НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

«На правах рукопису»
УДК 004.91

«До захисту допущено»
Науковий керівник кафедри
_____ І.А. Дичка
«___»_____2021 р.

Магістерська дисертація

на здобуття ступеня магістра

освітньо-науковою програмою
«Інженерія програмного забезпечення комп'ютерних та інформаційно-пошукових систем»

зі спеціальності 121 Інженерія програмного забезпечення

на тему: «Метод автоматизованого реферування україномовних науково-технічних текстів»

Виконала:
студентка ІІ курсу, групи КМ-91мн
Левчук Ольга Сергіївна _____

Керівник:
Доцент кафедри ПЗКС, к.т.н., доцент,
Заболотня Тетяна Миколаївна _____

Консультант з нормоконтролю:
Доцент кафедри ПЗКС, к.т.н., доцент,
Онай Микола Володимирович _____

Рецензент:
Доцент кафедри ПМА, к.ф.-м.н., доцент,
Костюшко Ірина Анатоліївна _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____

Київ – 2021 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

Рівень вищої освіти – другий (магістерський)

Спеціальність – 121 «Інженерія програмного забезпечення»

Освітньо-наукова програма «Інженерія програмного забезпечення комп'ютерних та інформаційно-пошукових систем»

ЗАТВЕРДЖУЮ

Науковий керівник кафедри

_____ Іван ДИЧКА

«___»_____2019 р.

ЗАВДАННЯ
на магістерську дисертацію студенту

Левчук Ользі Сергіївні

1. Тема дисертації «Метод автоматизованого реферування україномовних науково-технічних текстів», науковий керівник дисертації Заболотня Тетяна Миколаївна, к.т.н., доцент, затверджені наказом по університету від «26» березня 2021 р. №899-С.

2. Термін подання студентом дисертації «18» травня 2021 р.

3. Об'єкт дослідження: процес автоматизованого реферування текстових даних.

4. Предмет дослідження: методи та алгоритми автоматизованого реферування україномовних науково-технічних текстів.

5. Перелік завдань, які потрібно розробити:
   - проаналізувати існуючі методи автоматичного реферування тексту;
   - запропонувати метод автоматизованого реферування україномовних науково-технічних текстів;
   - програмно реалізувати метод автоматизованого реферування україномовних науково-технічних текстів на основі аналізу існуючих методів;
   - провести аналіз отриманих результатів.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу:
   - схема узагальненого алгоритму автоматичного реферування;
   - схема алгоритму визначення границь елементарних текстових елементів;
   - схема алгоритму побудови множини функціональних відношень;
   - компоненти розробленого програмного забезпечення;
   - результати роботи розробленого методу.

7. Орієнтовний перелік публікацій:
   - Тези доповіді "Абстрагуючий метод автоматизованого реферування україномовних науково-технічних текстів ".

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Онай М.В., доцент кафедри ПЗКС		

9. Дата видачі завдання «04» жовтня 2019 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1.	Ґрунтовне ознайомлення з предметною галуззю	15.10.2019	
2.	Визначення структури магістерської дисертації; вивчення літератури, пошук додаткової літератури, патентний пошук	01.12.2019	
3.	Робота над першим розділом магістерської дисертації; проведення наукового дослідження	01.03.2020	
4.	Проведення наукового дослідження; робота над другим розділом магістерської дисертації; розроблення програмного забезпечення	04.06.2020	
5.	Проведення наукового дослідження; робота над третім розділом магістерської дисертації; підготовка матеріалів доповіді на конференції ПМК-2020	20.01.2021	
6.	Завершення роботи над основною частиною магістерської дисертації; підготовка ілюстративного матеріалу;	01.03.2021	
7.	Оформлення текстової і графічної частини магістерської дисертації	25.04.2021	

Студент Ольга ЛЕВЧУК

Науковий керівник Тетяна ЗАБОЛОТНЯ

РЕФЕРАТ

Актуальність теми. З розвитком інформаційних ресурсів Інтернет обсяг електронної науково-технічної інформації значно збільшився, внаслідок чого пошук необхідної інформації значно ускладнився. В даній ситуації актуальним шляхом вирішення даної проблеми є методи автоматизованого реферування тексту, що допомагають отримати стисле представлення текстових документів – рефератів.

Однак, задача автоматичного реферування не є новою, над даною проблемою працювали багато дослідників понад 60 років. З самого початку роботи над вирішенням задачі автоматизованого реферування текстових даних, було винайдено декілька груп мето реферування тексту, а саме: екстрактивні, статистичні та методи з опорою на знання. Однак, текст за своєю природою має ієрархічну та нелінійну структуру, а перші два підходи вважають текст як лінійну послідовність слів та словосполучень – для отримання коректного реферату доцільним є використання останнього підходу, з опорою на знання. В результаті аналізу існуючих підходів, не існує готових програмних засобів, що базуються на методах автоматичного реферування тексту з опорою на знання для україномовних науково-технічних текстів, таким чином задача створення нового ефективного методу автоматизованого реферування україномовних науково-технічних текстів, що враховують нелінійну та ієрархічну структуру тексту є актуальною.

Об’єктом дослідження є процес автоматизованого реферування текстових даних.

Предметом дослідження є методи та алгоритми автоматизованого реферування україномовних науково-технічних текстів.

Мета роботи: покращення точності процесу автоматизованого реферування україномовних науково-технічних текстів шляхом розробки

та програмної реалізації нових методів та алгоритмів, що врахують нелінійну та ієрархічну природу тексту.

Методи дослідження. В роботі використовуються теорія риторичної структури, теорія предикатів, метод експертних оцінок та сучасні технології програмування.

Наукова новизна роботи полягає в наступному:

1. Запропоновано метод формалізованого опису структури тексту, що базується на використанні теорії риторичної структури, відрізняється врахуванням нелінійної та ієрархічної структури тексту для визначення критерію коректності тексту, що дозволяє підвищити точність отриманих рефератів.

2. Запропоновано алгоритм визначення функціональних відносин між фрагментами тексту з використанням спеціалізованого словника ключових фраз, що дозволяє зменшити надлишковість, що виникає при використанні словників загального призначення.

Практична цінність отриманих в роботі результатів полягає в тому, що розроблені методи та алгоритми дозволяють будувати системи автоматичного реферування україномовного науково-технічного тексту з врахуванням нелінійної та ієрархічної структури тексту, що дозволяє підвищити якість отриманих рефератів.

Апробація роботи. Основні положення і результати роботи були представлені та обговорювались на XIII науковій конференції магістрантів та аспірантів «Прикладна математика та комп'ютинг» ПМК-2020.

Структура та обсяг роботи. Магістерська дисертація складається з вступу, чотирьох розділів, висновків та додатків.

У вступі надано загальну характеристику роботи, описано актуальність задачі та проблеми, які вона може вирішити.

У першому розділі розглянуто основні методи автоматизованого реферування текстових даних, проаналізовано їх особливості та випадки

використання, в результаті чого до розробки був обраний підхід автоматичного реферування тексту на основі врахування особливостей тексту, оскільки було визначено, що статистичні методи аналізу тексту, на базі яких працюють існуючі системи автоматизованого реферування тексту, не враховують ієрархічність природномовних текстів.

У другому розділі розроблений та описаний метод формалізованого опису структури тексту, на якому базується метод автоматизованого реферування тексту.

У третьому розділі описані алгоритми, необхідні для автоматизованого реферування україномовних науково-технічних текстів, базуючись на розробленому методі.

У четвертому розділі була розглянута система автоматизованого реферування україномовного науково-технічного тексту та проведена оцінка ефективності розробленого методу та алгоритмів.

У висновках проаналізовано отримані результати роботи.

У додатках наведено реалізацію методу автоматизованого реферування тексту.

Робота виконана на 123 аркушах, містить 3 додатки та посилання на список використаних літературних джерел з 35 найменувань. У роботі наведено 13 рисунків та 15 таблиць.

Ключові слова: теорія риторичної структури, реферування, науково-технічні тексти, функціональні відношення.

ABSTRACT

Theme urgency. With the development of information resources on the Internet, the amount of electronic scientific and technical information has increased significantly, resulting in the search for the necessary information has become much more difficult. In this situation, the actual way to solve this problem is the automatic abstracting methods, which help obtain a concise presentation of text documents - abstracts.

However, the task of automatic abstracting is not new; many researchers have been working on this problem for over 60 years. From the beginning of the work on solving this problem, several abstracting approaches were invented: extractive, statistical, and methods based on knowledge. However, by its nature, the text has a hierarchical and nonlinear structure, and the first two approaches consider the text as a linear sequence of words and phrases - to obtain a correct abstract, it is advisable to use the latter approach, based on knowledge. As a result of the analysis of existing approaches, there are no ready-made systems based on methods of automatic abstracting of text based on knowledge for Ukrainian-language scientific and technical texts; thus the task of creating a new effective method of automated abstracting of Ukrainian-language scientific and technical texts, taking into account nonlinear the text is relevant.

Object of research is the process of automated abstracting of text data.

Subject of research are methods and algorithms of automated abstracting of Ukrainian-language scientific and technical texts.

Research objective: improving the accuracy of the process of automated abstracting of Ukrainian-language scientific and technical texts by developing and software implementation of new methods and algorithms that take into account the nonlinear and hierarchical nature of the text.

Research methods. The paper uses the theory of rhetorical structure, predicate theory, the method of expert evaluations and modern programming technologies.

Scientific novelty consists in the following:

1. A method of formalized description of the text structure, based on the use of rhetorical structure theory, is proposed to consider the nonlinear and hierarchical structure of the text, which includes determining the criterion of text correctness and many restrictions and characteristics for correct text structures.

2. The algorithm of the definition of functional relations between fragments of the text using the specialized dictionary of key phrases that allows reducing the redundancy arising at the use of dictionaries of general purpose is offered.

Practical value of the obtained results consists in the following: developed methods and algorithms allow building systems of automatic abstracting of the Ukrainian-language scientific and technical text, taking into account nonlinear and hierarchical structure of the text that allows improving quality of the received abstracts.

Approbation. The basic points and outcomes of the research have been presented and discussed at the XIII scientific conference for students and postgraduates «Applied mathematics and computing» PMK-2020.

Structure and content of the thesis. The master thesis consists of the introduction, four chapters, conclusions and appendixes.

The introduction provides a general description of the work, describes the relevance of the problem and the problems it can solve.

The first section considers the main methods of automated abstracting of text data, analyzes their features and uses, resulting in the development of an approach to automatic abstracting of text based on the features of the text, as it was determined that statistical methods of text analysis, based on existing automated text abstracting systems do not take into account the hierarchy of natural language texts.

The second section develops and describes a formalized description of the text structure, which is based on the method of automated text abstracting.

The third section describes the algorithms required for automated abstracting of Ukrainian-language scientific and technical texts based on the developed method.

In the fourth section, the system of automatic abstracting of the Ukrainian-language scientific and technical text was considered, and the efficiency of the developed method and algorithms was evaluated.

The results of the work are analyzed in the conclusions.

The appendices show the implementation of the method of automated text abstracting.

The work is done on 125 sheets, contains three appendices, and links to the list of used literature sources from 35 titles. The paper contains 13 figures and 15 tables.

Key words: theory of rhetorical structure, abstracting, scientific and technical texts, functional relations.

ЗМІСТ

ВСТУП

З розвитком інформаційних ресурсів Інтернет обсяг електронної науково-технічної інформації значно збільшився, внаслідок чого пошук необхідної інформації значно ускладнився. В даній ситуації актуальним шляхом вирішення даної проблеми є методи автоматизованого реферування тексту, що допомагають отримати стисле представлення текстових документів – рефератів.

Однак, задача автоматизованого реферування не є новою, над даною проблемою працювали багато дослідників понад 60 років, найвідомішими з яких є Г.П. Лун [1], І.П. Севбо [2], В.Є. Берзон [3], Е.Ф. Скороходько [4] та інші. З самого початку роботи над вирішенням даної проблему було винайдено декілька типів методів реферування тексту, а саме: екстрактивні, статистичні та методи з опорою на знання. Однак, текст за своєю природою має ієрархічну та нелінійну структуру, а перші два підходи вважають текст як лінійну послідовність слів та словосполучень. Тому для отримання реферату, що врахує особливості структури тексту, доцільним є використання останнього підходу, з опорою на знання. В результаті аналізу існуючих програмних засобів, не існує готових систем, що базуються на методах автоматизованого реферування тексту з опорою на знання для україномовних науково-технічних текстів. Таким чином, задача створення нового методу автоматизованого реферування україномовних науково-технічних текстів, що враховує нелінійну та ієрархічну структуру тексту є актуальною.

1. АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ АВТОМАТИЗОВАНОГО РЕФЕРУВАННЯ ПРИРОДНОМОВНИХ ТЕКСТОВИХ ДАНИХ

1.1. Задача автоматизованого реферування тексту

Однією з найбільш важливих проблем, що виникає у даний час при необхідності роботи та аналізу великого обсягу документів є проблема мінімізації часу на обробку кожного з них. Зрозуміло, що чим меншим є обсяг документу, то і час, що є необхідним для його аналізу також є меншим. Саме тому в тексті особлива увага приділена задачі автоматизованого аналізу текстових документів з метою виділення смислових одиниць, що будуть стисло описувати матеріал.

В подальшому під терміном реферату будемо мати на увазі згенерований стислий текст, що висловлює основну думку обраного документу. За допомогою реферату, в його ретельно відібраній структурі, користувач зможе проводити пошук необхідних фактів набагато швидше та якісніше. Необхідність рефератів виникає в таких галузях, як пошукові системи, аналітичні звіти та генерація стислого тексту для його подальшого перекладу.

Під автоматизованим реферуванням тексту мається на увазі створення стислого реферату комп’ютером, отже необхідною умовою є наявність алгоритму, що буде приймати у якості вхідних даних тексти, з яких необхідно згенерувати реферати, а на виході – отримані реферати. Вхідні тексти характеризуються наступними параметрами:

- Кількість документів (можливими значеннями даного параметру є: один чи кілька документів).
- Специфіка тексту (можливими значеннями даного параметру є: специфічна предметна область чи документ загального призначення).
- Перелік мов (можливими значеннями даного параметру є: одна мова чи декілька).

Отримані реферати характеризуються наступними параметрами:

- Наявність логічності та послідовності тексту.
- Ступінь інформативності.
- Ступінь стиснення.
- Жанр тексту (можливими значеннями даного параметру є: науково-технічні звіти, новини, книги та інші).
- Форма подання (можливими значеннями даного параметру є: табличне подання, текстове або у формі списку).
- Стиль тексту.

Вирішення задачі автоматизованого реферування текстових даних розпочалось понад двадцять років тому, майже з початком розвитку обчислювальної техніки та досі не закінчилось, оскільки прийнятного результату вирішення більшості практичних задач досі немає. Одним з найбільш важливих напрямків є пошук оптимальних шляхів та методів автоматизованого стиснення документів. Під терміном стиснення документу будемо розуміти сукупність операцій обробки інформації з метою зменшити її обсяг та максимально зберегти інформативність.

Методи загального призначення не можуть забезпечити достатньої інформативності для реферування загального класу текстів, єдиним шляхом є розбиття класу текстів загального призначення на сукупність підкласів, де тексти одного підкласу мають схожі властивості. До того ж, за кожним підкласом текстів закріплений свій тип літературного мовлення, що мають специфічні властивості. Для української мови можна виділити наступні стилі:

- художній;
- офіційно-діловий;
- науковий;
- науково-технічний;
- публіцистичний.

Розглянемо особливості наукового та науково-технічного стилів.

Науковий стиль характеризується великою кількістю термінів а також логічністю та послідовністю викладення матеріалу. Організація тексту характеризується тим, що кожен фрагмент тексту, що є логічно завершеним, містить вступні та заключні речення, які містять стислу основну інформацію. Саме цю особливість наукових текстів є доцільним враховувати для реферування документів даного стилю.

Науково-технічний текст відрізняється за наукові особливим формально-логічним способом викладення матеріалу [5]. Типове викладення тексту представляє собою міркування, метою якого є опис та доказ наведених досліджень. В текстах даного типу найбільш розповсюдженими конструкціями, що описують міркування автора, є: «внаслідок», «спершу», «нарешті», «є підстави вважати» та інші. Слова та словесні конструкції даного типу називають дискурсивними маркерами, оскільки вони виконують важливу функцію передачі намірів автора довести та описати проведене наукове дослідження [6]. Найбільш типовими дискурсивними операціями є приведення фактів, гіпотез, висновок та його обґрунтування та інші. У більшості випадків дані операції мають свої загальнонаукові дискурсивні мітки. Приведемо найбільш розповсюджені типи зв’язків між елементами тексту:

- опис та констатація;
- уточнення та введення додаткової інформації;
- причинно-наслідкові зв’язки та логічні операції;
- актуалізація уваги, що полягає у підкресленні інформації;
- допущення та визначення;
- приведення прикладів:
- підведення підсумків;
- цитування та приведення прикладів;
- оцінювання та висловлення власної думки.

До методів організації та структуризації наукового тексту можна віднести нумерацію, розбиття на розділи або підрозділи та абзацний поділ. До того ж, особливостями науково-технічних текстів є:

- відсутність естетичної функції;
- однозначність сприйняття;
- прозорість, точність і повнота змісту.

1.2. Огляд основних методів автоматизованого реферування тексту

Початок роботи над методами автоматизованого реферування тексту розпочався майже з самого початку активного використання електронно-обчислювальною технікою. Перший експеримент для вирішення задачі автоматизованого реферування текстових даних був проведений в 1957 році в США [7]. На відміну від автоматизованого перекладу, на перших його етапах, де реалізація методів проводилась «фраза за фразою» [8], в задачі автоматизованого реферування текстів основна увага була зосереджена на більші за обсягом ділянки тексту, ніж речення, в яких містилось міркування на схожу тему. Тобто, дослідники з самого початку зосереджувались на пошуку закономірностей, що організують смислову єдність тексту [9].

На першому етапі наукових робіт з автоматизованого реферування текстових даних найбільш популярними були методи, що базувались на визначенні статистичних закономірностей появи термінів або їх комбінацій у тексті [10]. В подальшому дослідження в даній області змінились на дослідження та використання внутрішніх структур тексту та інформаційної основи, на якій будується весь текст [11].

На сьогодні можна відокремити дві основних групи методів до задачі автоматизованого реферування тексту:

- екстрактивні методи;
- методи з опорою на знання.

Методи, що базуються на першій групі методів, полягають у виділенні окремих блоків тексту, що мають найбільшу лексичну та статистичну

релевантність, з яких будується підсумковий документ. У більшості методів даного типу використовується модель лінійних вагових коефіцієнтів, яка полягає в тому, що кожному блоку тексту призначуються вагові коефіцієнти відповідно до їх розташування у тексті та статистичній значущості.

Методи, що базуються на другому підході, містять в собі три етапи: аналіз тексту та побудова його формального опису, після чого з формального опису відбувається вибірка ключових моментів, з яких формується реферат.

В рамках екстрактивних методів є відомими декілька груп методів:

- Статистичні методи, що базуються на використанні статистичних параметрів для оцінки інформативності різноманітних елементів тексту, таких як слова або речення. Перш за все, таким статистичним параметром є частота появи слів в тексті: на основі результатів ранжування лексики в тексті дані методи визначають слова з високим рангом та частоту їх появи у різних словосполученнях, після чого, згідно з отриманими результатами, визначається інформативність фраз.
- Позиційні методи, що базуються на припущенні, що інформативність речення залежить від його позиції у документі. Проте, даний підхід може працювати лише на ретельно структурованих документів, таких як стандарти або патентні описи, у всіх інших типах документів отримані реферати не будуть репрезентативними.

З наведених екстрактивних методів автоматизованого реферування текстів тільки статистичні дійшли до практичної реалізації у промислових цілях.

Таким чином, для сучасних методів є характерним модифікація традиційних груп методів. Наприклад, визначаються додаткові критерії відбору значущих слів, таких як підвищення значущості слова в залежності

від його положення у заголовку, перших або останніх реченнях або стилю форматування (виділення шрифтами в тексті) [12].

Існує також метод симетричного реферування [13], в якому розрахунок ваги речення здійснюється за допомогою кількості зв’язків між поточним реченням та реченнями, що знаходяться з обох сторін від нього. Для реалізації даного методу в кожному реченні визначається список ключових слів, що містяться в заздалегідь підготовленому словнику ключових слів, після чого, в реченнях, з якими визначаються зв’язки підраховується кількість найдених в них ключових слів з списку, що був побудований раніше з поточного речення. Кількість даних зв’язків і визначає вагу речення.

Реферування тексту за допомогою других груп методів, методів з опорою на знання, використовує більш складні лінгвістичні алгоритми, але при цьому, в результаті роботи генерується не лише набір речень вхідного файлу, а породжується новий текст, що є рефератом, який змістовно узагальнює вхідний документ. В цьому випадку, для підготовки стислого викладу тексту є необхідними потужні обчислювальні ресурси для систем обробки природних мов та граматики зі словниками, що використовуються у синтаксичному розборі та генерації конструкцій обраною природною мовою. Окрім того, необхідними є словники, орієнтовані на предметну область, для прийняття рішень під час аналізу тексту для визначення найбільш важливої інформації. В даному типі методів на основі лінгвістичного методу синтаксичного розбору речень будуються дерева розбору, частини яких перегруповують та видаляють гілки, що є менш значущими, згідно структури тексту (наприклад, дужок). Після даних спрощень, дерево розбору значно спрощується і являє собою стиснення початкового тексту. Розглянемо на рисунку 1.1, схему роботи даного типу методів:

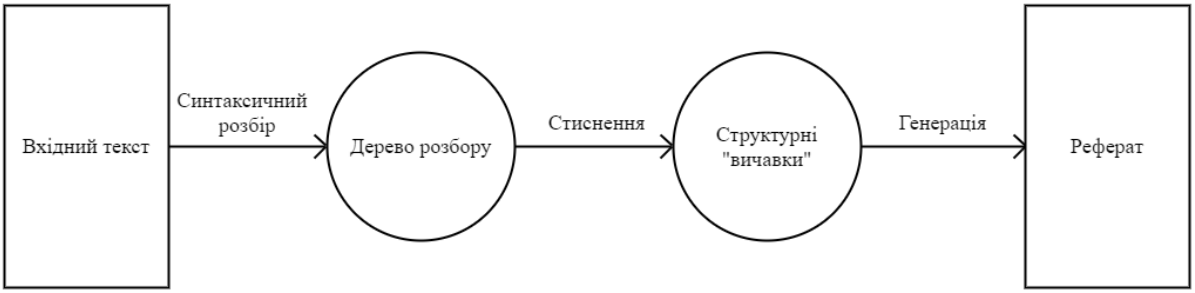


Рис. 1.1. Схема роботи методів з опорою на знання

Під час побудови реферату початкове представлення тексту набуває значних змін. Надлишкова інформація видаляється шляхом видалення підграфів, що містять поверхневі судження. Далі отримані графи поєднуються між собою, в результаті чого формується концептуальна репрезентативна структура реферату, що являє собою смислове стиснення початкового тексту.

Даний тип методів потребують значних обчислювальних ресурсів для систем обробки природних мов , різноманітних словників та баз знань та на сьогоднішній день не мають повністю закінчених програмних реалізацій.

Отже, підведемо підсумки з аналізу існуючих методів.

Екстрактивні методи досить легко налаштувати для обробки великих обсягів інформації. Вони є простими до реалізації, однак нелінійність структури тексту не враховується, тому текст реферату може бути незв’язним відносно тексту, що подається у якості вхідних даних.

Методи з опорою на знання є більш складними та отримані за їх допомогою реферати досить часто містять інформацію, що доповнює основний текст. Оскільки вони спираються на формальне представлення структури документу, їх можливо налаштувати на досить високу ступінь стиснення. Дані методи у більшості випадків потребують словників та потужного обладнання, що є перешкодою для широкого використання даного типу методів.

1.3. Сучасні системи автоматизованого реферування тексту

На сьогоднішній день багато відомих виробників програмного забезпечення пропонують свої програмні рішення для автоматизованого реферування тексту. Розглянемо деякі з них:

- Intelligent Miner for Text;
- MS Office AutoSummarize;
- TextAnalyst;
- Oracle Text;
- Copernic summarizer.

Intelligent Miner for Text – є продуктом фірми IBM, що містить в собі кілька утиліт, що запускаються з командного рядка та працюють незалежно один від одного. Дана система є однією з найкращих для аналізу текстів та одна з його компонент під назвою Annotation Tool призначена для створення рефератів [14].

MS Office AutoSummarize – вбудована в програму Microsoft Word функція автоматизованого отримання реферату [15].

TextAnalyst – є продуктом фірми Microsystems, що вирішує задачу глибокого аналізу тексту, наприклад підготовка резюме тексту, що у більшості випадків є достатнім для отримання уявлення про текст [16].

Oracle Text – є продуктом фірми Oracle, що є частиною Oracle Database, системи управління реляційними базами даних. Він може аналізувати текст та генерувати його резюме [17].

Copernic summarizer – використовує складні статистичні алгоритми для пошуку найбільш значущих речень, не має обмежень за тематикою та генерує реферат згідно вимог користувача [18].

Приведемо порівняльну характеристику наведених систем в табл. 1.1.

Таблиця 1.1

Порівняльна характеристика систем автоматизованого реферування

Назва	Метод реферування	Підтримка української мови
Intelligent Miner for Text	Статистичні методи	Відсутня
MS Office Auto-Summarize		Присутня
TextAnalyst		Відсутня
Oracle Text		Присутня лише при завантаженні словників власноруч
Copernic summarizer	Статистичні та лінгвістичні методи	Відсутня

На основі огляду існуючих систем, можна стверджувати, що автоматизоване реферування україномовних текстів присутнє лише за допомогою статистичних методів, що не враховують нелінійну природу тексту.

1.4. Головні проблеми формалізації структури тексту

На сьогоднішній день проводиться безліч досліджень в галузі формалізації текстових даних, які засновані на досягненнях в галузі комп'ютерної лінгвістики. Результати даних досліджень використовуються в автоматизації процесів обробки інформації.

Розглянемо спільні положення існуючим теоріям, що досліджують структуру тексту [19].

Перше положення: елементарні текстові елементи являють собою частини тексту, що не перетинаються. Деякі дослідники вважають елементарними частинами тексту слова або словосполучення [20], в той час як інші дослідники вважають елементарними текстовими елементами речення [21].

Друге положення: відношення знаходяться між елементами різного розміру. Існує декілька досліджень щодо природи та кількості

функціональних відношень, що лежать між елементами тексту. В одному з них результатом став повний набір відношень, що базуються на синтаксичних відношеннях між частинами речень [20]. В іншому доводиться точка зору, що класифікація функціональних відношень повинна бути базуватися на чітко визначених принципах, які виводяться з лексикографічних принципів, типів висновків, які є необхідними читачу для розуміння тексту, результатів, які намагався донести автор, ключових фраз [22]. У даних дослідженнях є спільна риса, а саме припущення, що відношення повинні доповнювати сенс тексту.

Третє положення: деякі текстові елементи є більш значущими за інші. Дане положення базується на існуванні двох типів функціональних відношень: симетричних та асиметричних. Симетричне відношення також називають «багатоядерним», бо воно не має чітко визначеного ядра, тобто незалежної частини. В свою чергу, асиметричне відношення має чітко визначену незалежну та більш значущу частину, що називається ядром, та залежну, що називається сателітом.

Четверте положення: структура тексту може бути представленою у вигляді дерева. Більшість теорій стверджують, но найбільш оптимальною математичною структурою тексту є саме дерево, де елементарні текстові елементи є листками дерева [23].

Таким чином, для вирішення задачі автоматизованого реферування україномовних науково-технічних текстів, необхідно вирішити задачу формалізації структури тексту, що полягає у вирішенні проблеми визначення специфікації обмежень, що характеризують коректну структуру тексту.

1.5. Постановка задачі дослідження

На сьогоднішній день робота з природномовним текстом з використанням обчислювальної техніки є актуальною задачею для дослідження, оскільки більшість систем використовують статистичні

методи, що не враховують нелінійну природу тексту, що значно погіршує ступінь донесення тексту до читача.

Структура тексту визначається особливостями внутрішньої організації текстових елементів та їх взаємозв'язків в рамках тексту як одиниці спілкування. Кожен текст має свої власні ознаки, що визначають його стиль.

В даній роботі пропонується підхід вирішення задачі автоматизованого реферування україномовних науково-технічних текстів з врахуванням нелінійної природи тексту, що допоможе більш якісно вирішити задачу автоматизованого реферування україномовних науково-технічних текстів, ніж існуючі статистичні методи. В даній роботі основна увага буде приділятися питанням опису структури тексту та алгоритмам автоматизованого аналізу та побудови структури тексту для вирішення задачі автоматизованого реферування україномовних науково-технічних текстів.

Метою даної роботи покращення точності процесу автоматизованого реферування україномовних науково-технічних текстів шляхом розробки та програмної реалізації нових методів та алгоритмів, що врахують нелінійну та ієрархічну природу тексту. Для досягнення даної мети, необхідно: проаналізувати існуючі методи реферування тексту; розробити метод автоматизованого реферування україномовних науково-технічних текстів; розробити систему автоматизованого реферування україномовних науково-технічних текстів та оцінити результати роботи.

1.6. Висновки до першого розділу

На основі аналізу існуючих підходів автоматизованого реферування текстів, можна зробити висновок, що методи з опорою на знання майже не використовуються, оскільки недостатньо досліджені для опису властивостей тексту.

Аналіз існуючих систем автоматизованого реферування україномовних текстів показав існування тільки вбудованої в програмний продукт Microsoft Office функцію Autosummarize, яка базується на статистичних методах і не враховує нелінійну та ієрархічну природу тексту, що може призвести до значної втрати інформації.

В результаті проведеного аналізу було сформульовано мету та основні задачі роботи.

2. МЕТОД ФОРМАЛІЗОВАНОГО ОПИСУ СТРУКТУРИ ТЕКСТУ

2.1. Підходи до опису структури тексту

Опис тексту у запропонованому методі базується на теорії риторичної структури, в рамках якої пропонується універсальний метод опису семантичної структури тексту у вигляді графу, який надасть можливість опису структури тексту за допомогою скінченного набору семантичних зв'язків. Дані, подані в такому форматі, можуть бути використані для порівняльного аналізу структури текстів, забезпечуючи при цьому отримання статистично достовірних результатів. Згідно даної теорії, будь-який текст може бути поданий у вигляді графа, вузлами якої є елементарні дискурсивні одиниці або їх групи – дискурсивні одиниці. При цьому, вузли графів будуть зв'язані один і тим самим набором відношень, які називаються функціональними відношеннями, що означає, що кожна дискурсивна одиниця не існує самостійно, а додається до іншої для досягнення певної мети.

Кожна дискурсивна одиниця, що належить до функціонального відношення, може відігравати в ньому різну роль, більш значущий компонент функціонального відношення називається ядром, а менш значущий – сателітом. При цьому сателіт, на відміну від ядра, у більшості випадків може бути пропущеним або заміненим на інший при збереженні функціонального ефекту. У більшості випадків відношення асиметричні, тобто містять ядро і сателіт. Наведемо приклади відношень в таблиці 2.1.

Таблиця 2.1

Приклади функціональних відношень

Відношення	Ядро	Сателіт
sentence	Перший елемент в перерахуванні	Другий елемент в перерахуванні
background	Текст, необхідний для розуміння	Текст для полегшення розуміння
elaboration	Базова інформація	Додаткова інформація

Якщо відношення не має сателіту, то його називають багатоядерним. Наведемо приклад в таблиці 2.2.

Таблиця 2.2

Приклад багатоядерного функціонального відношення

Відношення	Ядро	Друге ядро
contrast	Перша альтернатива	Друга альтернатива

Використовуючи функціональні відношення, можна виконувати аналіз тексту, який полягає в побудові структури тексту, що полегшить його розуміння читачем.

Зв'язність тексту, згідно теорії риторичних структур, полягає в сукупності обмежень на функціональні відношення. Обмеження накладаються на ядро, сателіт та їх комбінації. Приведемо приклад, розглянувши функціональне відношення, що описує очевидний факт між елементами А та В. При цьому ядро А представляє собою сам факт, а сателіт В – додаткову інформацію, яка необхідна, щоб читач впевнився в достовірності даного факту. З цього слідує, що без сателіту читач може не довіряти інформації, викладеної в ядрі. Отже, з цього слідують наступні обмеження:

Обмеження на ядро – читач може не довіряти інформації у тій степені, як її хотів донести автор

Обмеження на сателіт – читач довіряє даній інформації

Обмеження на комбінацію – отримання інформації з сателіту підвищує довіру до інформації до ядра

При побудові дискурсного дерева з функціональних відношень, необхідно використовувати основні положення теорії риторичної структури:

- елементарні текстові елементи є непересічними між собою частинами тексту;

- функціональні відношення поєднують між собою частини тексту різного розміру;
- елементарні текстові елементи мають різну значущість у тексті;
- структура тексту може бути представленою у вигляді дерева;

Також, теорія риторичних структур накладає наступні обмеження на структуру тексту для визначення її коректності:

- функціональні структури є деревами, в яких елементи, що знаходяться на одному рівні являють собою неперервний текст;
- елементи можуть бути двох видів: ядро та сателіт;
- кожен текстовий елемент може бути пов’язаний з іншим лише одним функціональним відношенням;

Але незважаючи на викладенні в теорії риторичних структур обмеження, використання її для автоматизації процесу побудови структури тексту є складним через наступні недоліки:

- не існує формальної специфікації, яка дозволила б визначити некоректне дерево;
- нема алгоритмів побудови дерев, що описують структуру тексту.

Отже, метод формалізації опису структури тексту, запропонований в даній роботі, включає в себе:

- визначення критерію коректності структури тексту;
- визначення характеристик структури тексту;
- визначення обмежень на коректні структури тексту;

2.2. Розробка критерію коректності структури тексту

В теорії риторичної структури нема правил щодо об'єднання двох суміжних фрагментів тексту в один. Але оскільки фрагмент, що є ядром є більш значущим, ніж фрагмент-сателіт, то можна зробити висновок що при видаленні фрагментів, що є ядрами – сенс тексту зникне.

На основі аналізу україномовних науково-технічних текстів висувається припущення, що функціональні відношення, що знаходяться

між великими фрагментами тексту та являють собою великі дерева, можуть бути зумовлені існуванням функціональних відношень між піддеревами, що існували до їх об'єднання у більше дерево.Дане припущення було зроблене на базі того, що після аналізу різноманітних україномовних науково-технічних текстів будо виявлено, що замість того, щоб одразу оцінювати, яке функціональне відношення знаходиться між двома великими фрагментами тексту, було набагато простіше асоціювати деякі абстракції з даними фрагментами і тільки після цього визначати відношення між абстракціями. Дані унікальні абстракції у більшості випадків відносились до фрагментів текстів, що були ядрами.

Таким чином, на основі аналізу україномовних науково-технічних текстів,а також робіт [23, 24], висувається наступне обмеження для визначення коректності структури тексту: якщо функціональне відношення лежить між двома елементами структури тексту, то воно лежить між, принаймні, двох ключових складових цих елементів. Основною ідеєю даного критерію є те, що оскільки ядра є більш значущими, ніж сателіти, то основний сенс тексту має зберегтись, якщо видалити всі сателіти.

2.3. Особливості представлення структури тексту

Для опису коректної структури тексту припускається, що функціональні відношення, що знаходяться між великими текстовими елементами, повинні буди пояснені за допомогою функціональних відношень між елементарними текстовими елементами, які можуть бути визначені однозначно.Розглянемо детальніше вказані припущення.

Перше припущення: є необхідність роботи з розширеними функціональними відношеннями. Критерій коректності для структури тексту пояснює наявність розширених функціональних відношень через функціональні відношення між елементарними текстовими елементами, але в деяких випадках можливо визначити функціональні відношення між більш великими текстовими елементами, без необхідності відокремлювати

елементарні текстові елементи. Наприклад, фрагмент тексту може складатись з трьох параграфів, що об'єднані функціональним відношенням «послідовність». В цьому випадку можливо побудувати структуру тексту без відокремлення важливих елементів даних параграфів.

Друге припущення: є необхідність обробки невизначеностей. Для визначення критерія коректності тексту припускається, що процес визначення функціональних відношень для елементарних текстових елементів є однозначним, що не зовсім так, оскільки в деяких випадках може бути складно одразу визначити тип відношення. Приведемо приклад на рис. 2.1.

[Перша система призначена для обробки векторної графіки.]^{A}[Найкраще вона працює з зображеннями формату SVG.]^{B}[На відміну від першої системи, друга призначена для більшої кількості форматів.]^{C}[Особливо добре вона працює з форматами PNGта JPG.]^{D}

Рис. 2.1. Приклад тексту

Даний текст розбитий на 4 блоки, між якими необхідно встановити функціональні відношення. З першого погляду, здається зрозумілим, що це є відношення «відмінність», оскільки присутнє ключове словосполучення «на відміну від», але не є зрозумілим, між якими елементами лежить дане відношення, між [A, B] та [C, D],або між A та [C, D], або між [A, B] та C, або між A та C? Найкращим шляхом вирішення даної проблеми є використання взаємовиключних функціональних відношень. Оскільки критерій коректної структури тексту дає можливість описати функціональні відношення між великими блоками тексту через функціональні відношення між елементарними текстовими елементами, то для наведеного тексту можна визначити наступну множину:

$$isInFunctionalRel(contrast, A, C) \oplus isInFunctionalRel(contrast, A, D) \oplus isInFunctionalRel(contrast, B, C) \oplus isInFunctionalRel(contrast, B, D)$$

Таким чином, будемо використовувати множину взаємовиключних функціональних відношень. Для наведеного тексту комп'ютерна програма може визначити множину функціональних відношень, використовуючи лише інформацію про ключові слова та зв'язність тексту. Функціональне відношення «удосконалення» між A та B виникає через зв'язність тексту, оскільки в обидвох реченнях мова йде про одну й ту ж саму систему. Таке ж функціональне відношення можна встановити завдяки ключовому слову «особливо».

Дані допущення дозволяють математично описати структуру тексту.

2.4. Побудова математичного опису структури тексту

Формалізація текстових структур передбачає наявність множини R текстових відношень, яке розподілено на дві подмножини: множину асиметричних відношень (з одним ядром та одним сателітом) та симетричних (з декількома ядрами). Отже:

$$R = R_{\text{симетричні}} \cup R_{\text{асиметричні}}.$$

Також, вона передбачає, що два суміжних текстових елементи можуть бути поєднані в один тільки тоді, якщо будуть виконані наступні умови:

- Таке ж відношення є між двома найважливішими частинами цих елементів.
- Між двома суміжними елементами тексту є розширене функціональне відношення.

Введемо наступні предикати для опису характеристик коректної структури:

- $isValidPosition(u_i, j)$, який є вірним якщо в послідовності U тільки якщо u_i є j-м елементом послідовності.
- $isInFunctionalRel(name, u_i, u_j)$, який є вірним для u_i та u_j по відношенню до функціонального відношення $name$, тільки якщо визначення D функціонального відношення $name$ узгоджено з

відношенням між u_i (у більшості випадків сателітом) та u_j (у більшості випадків ядром).

- $isBlockInFunctionalRel(name, s_s, s_e, n_s, n_e)$, який є вірним для фрагментів тексту [s_s, s_e] та [n_s, n_e] по відношенню до функціонального відношення name узгоджено з відношенням між фрагментами тексту, які покривають $[s_s,\ s_e]$ (у більшості випадків сателіт) та $[n_s, n_e]$ (у більшості випадків ядра).

В якості прикладу опису тексту розглянемо рис. 2.2.

[Неважливо, наскільки складною є будь-яка програмна система,]^A [контроль якості є дуже важливою частиною її розробки.]^B [Відомо, що дуже велика кількість помилок припускається на етапі проектування,]^C [хоча розробники вважають, що всі вимоги до системи є зрозумілими.]^D

Рис. 2.2. Приклад тексту

Описати даний текст можемо наступним чином:

$$\begin{cases} isInFunctionalRel(justification, \text{A}, \text{B}) \\ isInFunctionalRel(justification, \text{D}, \text{B}) \\ isInFunctionalRel(evidence, \text{A}, \text{B}) \\ isInFunctionalRel(concession, \text{D}, \text{C}) \\ isInFunctionalRel(restatement, \text{D}, \text{A}) \\ isValidPosition(\text{A}, 1) \\ isValidPosition(\text{B}, 2) \\ isValidPosition(\text{C}, 3) \\ isValidPosition(\text{D}, 4) \end{cases}$$

В якості прикладу представимо даний текст у вигляді бінарного дерева на рис. 2.1.

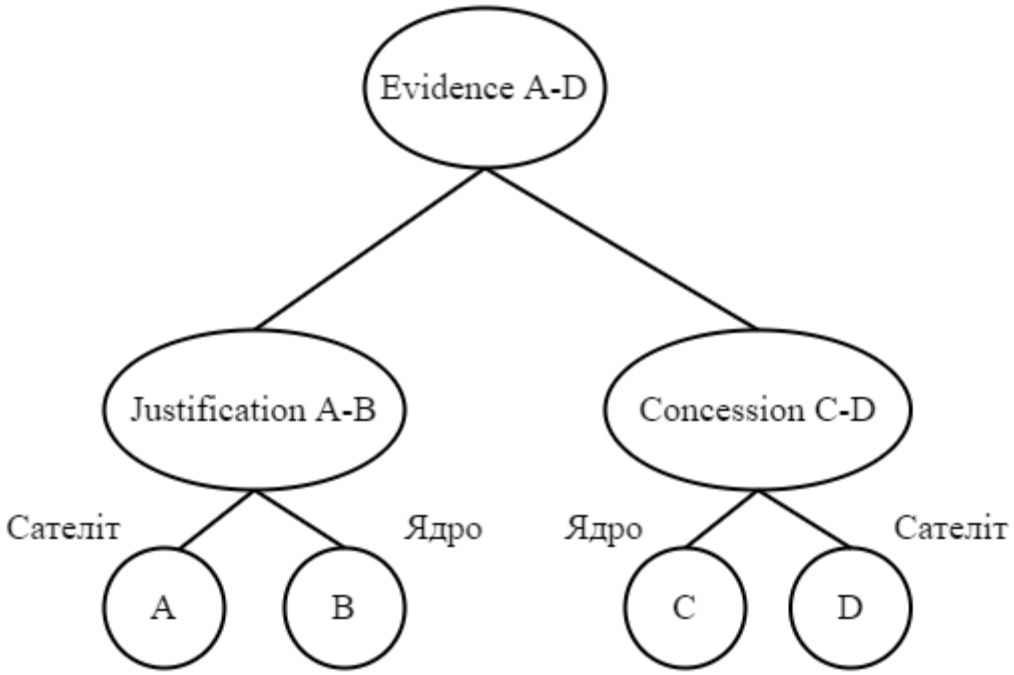


Рис. 2.3. Структура тексту

Отже, формалізація структури тексту побудована на наступних принципах:

- Як відомо з теорії риторичної структури, текст може бути представлений у вигляді бінарного дерева, листами якого є елементарні текстові елементи.
- Пропонується характеризувати кожен вузол наступними параметрами: $status$ (ядро чи сателіт), $type$ (функціональне відношення, яке лежить між його потомками), $valuable$ (множина найбільш важливих елементів цього вузла).

Статус, тип та кількість важливих нащадків, які є атрибутами кожного вузла, дають достатню кількість інформації для повного опису текстової структури. На етапі розгляду тексту заздалегідь ми не маємо можливості припустити, де будуть межі текстових фрагментів, тому нам необхідно визначити процедуру, яка полягає у переборі всіх можливих способів побудови різноманітних дерев з цієї лінійної послідовності тексту.

Нехай $block_{i,j}$ або $[i,j]$ є множиною всіх елементірних текстових елементів між i та j. Для послідовності елементарних текстових елементів

$u_1 u_2 \dots u_n$ існує n спосіб побудови фрагментів тексту довжиною в один елементарний текстовий елемент: $block_{1,1}, block_{2,2} \dots block_{n,n}$; n-1 спосіб побудови фрагменти тексту довжиною в два текстових елемента та один спосіб побудови фрагменту тексту довжиною n. Тому заздалегідь визначити ті фрагменту, що увійдуть в результуюче дерево, неможливо. Відповідно, зв'язав з кожним із них статус, тип та кількість ключових елементів, можна побудувати безліч коректних дерев, згідно критерію коректної структури тексту.

Для визначення обмежень на коректну структуру тексту пропонується використовувати наступні властивості для фрагменту $block_{i,j}$:

- $S(i, j, status)$ є вірним, якщо $block_{i,j}$ має статус $status$ (може приймати значення «ядро», «сателіт» чи «не визначено»).
- $T(i, j, type\_name)$ є вірним, якщо функціональне відношення, що лежить між прямим потомками $block_{i,j}$ в дереві співпадає з type_name.
- $V(i, j, valuable\_name)$ є вірним, якщо $valuable\_name$ входить до списку найбільш важливих елементарних текстових елементів для $block_{i,j}$.

Використовуючи введені вище властивості, можна математично описати перелік обмежень, необхідних для коректної структури тексту. Нехай N є кількістю елементарних текстових елементів в тексті.

- Для кожного фрагменту $[i, j]$ предикат S має наступні можливі значення: «ядро», «сателіт» або «не визначено». Для випадку, коли $i = j$ можливі значення лише «ядро» та «сателіт».

$$[(1 \le i \le N) \cap (1 \le j \le i)] \to \{[i = j \to \big(S(i,j,\text{ядро}) \cup S(i,j,\text{сателіт})\big)] \cap [i \ne j \to \big(S(i,j,\text{ядро}) \cup S(i,j,\text{сателіт}) \cup S(i,j,\text{none})\big)]\}.$$

- Статус будь-якого фрагменту тексту є унікальним.

$$[(1 \le i \le N) \cap (1 \le j \le i)] \to [(S(i,j,status_1) \cap S(i,j,status_2)) \to status_1 = status_2]$$

- Для кожного фрагменту $[i, j]$ предикату T можливими значеннями є множина функціональних відношень, що відповідають даному фрагменту.

$$[(1 \leq i \ \leq N) \cap (1 \leq j \leq i)] \rightarrow \{[i = j \rightarrow T(i,j,\text{leaf})]$$
$$\cap [i \neq j \rightarrow \big(T(i,j,\text{leaf}) \cup T(i,j,\text{name}) \rightarrow GetAll(i,j,name)\big)]\}$$

, де $GetAll(i,j,name)$ повертає множину функціональних відношень, що лежать між фрагментами тексту в $[i, j]$.

- Принаймні одне функціональне відношення знаходиться між двома суміжними фрагментами.

$$[(1 \leq j \leq N) \cap (1 \leq j \leq i)] \rightarrow [(T(i,j,name_1) \cap T(i,j,name_2)) \rightarrow name_1 = name_2]$$

- Для кожного фрагменту $[i,j]$ предикату V можливими значеннями є множина елементарних текстових елементів, з яких він складається.

$$[(1 \leq j \leq N) \cap (1 \leq j \leq i)] \rightarrow [V(i,j,NONE) \cup V(i,j,u) \rightarrow GetAll(i,j,u)]$$

Де $GetAll(i,j,u)$ повертає множину відношень, що лежать між елементами тексту в $[i, j]$.

- Текстовий фрагмент зі статусом «*none*» або «не визначено» не може бути присутнім у результуючому дереві.
- Існує головний фрагмент, який є коренем дерева та покриває увесь текст.

2.5. Висновки до другого розділу

У даному розділі розроблений метод формалізованого опису структури тексту, що базується на використанні теорії риторичних структур, яка допоможе врахувати нелінійну та ієрархічну природу тексту, що дозволяє підвищити якість автоматизованого реферування україномовних науково-технічних текстів. Метод формалізованого опису містить визначення характеристик структур тексту та обмежень, що накладаються на коректні структури тексту.

Також, було показано, що теорія риторичних структур дозволяє описати текст за допомогою ієрархічної структури, вузли якої пов’язані між собою функціональними відношеннями, які визначають значущість фрагментів тексту. Ця інформація може бути використана для генерації реферату.

Встановлено, що безпосереднє використання теорії риторичної структури для автоматизованого реферування україномовних науково-технічних текстів є неможливим через відсутність чіткої специфікації структур тексту, як і алгоритмів побудови таких структур, тому дану теорію необхідно доповнити.

Розроблений критерій коректності структури тексту, виконана формалізація обмежень та характеристик для коректних структур, що є доповненням до теорії риторичних структур. Вони визначають умови об’єднання фрагментів тексту, дозволяють мінімізувати набір необхідних параметрів, що є достатніми для повного опису структури тексту.

3. ЗАПРОПОНОВАНИЙ МЕТОД АВТОМАТИЗОВАНОГО РЕФЕРУВАННЯ ТЕКСТУ

3.1. Узагальнений алгоритм автоматизованого реферування тексту

В другій главі був приведений математичний опис коректних структур тексту, що являють собою набір властивостей, що задовольняє критерію коректності тексту, а саме: якщо функціональне відношення лежить між двома текстовими елементами, то це відношення лежить принаймні між двох елементарних текстових елементів, що є потомками цих елементів. Основний сенс даного критерію полягає в тому, що елементарні текстові елементи, що є ядрами є більш значущими за елементарні текстові елементи, що є сателітами, звідки слідує, що основний сенс тексту має зберегтись, якщо видалити всі сателіти. Якщо застосувати даний метод рекурсивно по всьому тексту, що представлений у формі дереву, мі отримаємо дерево, що задовольняє критерій коректності.

Також, при побудові дерева виникає необхідність в вирішенні невизначеності, що виникає в задачі пошуку функціональних відношень. Для вирішення даної задачі використовується множина функціональних відношень, що містить множину всіх можливих відношень, при чому передбачається, що в процесі побудови структури тексту кожне відношення може бути використано лише раз.

Алгоритм автоматизованого реферування тексту передбачає деякий вхідний текст та шляхом побудови його структури генерує реферат. Оскільки текст є ієрархічною структурою, що складається з параграфів, речень та словосполучень, алгоритм має враховувати це в процесі аналізу тексту.

Встановлено, що структура тексту є деревом, вузли якого є фрагментами тексту та пов’язані між собою функціональними відношеннями. Тому для побудови дерева, перш за все необхідно визначити ці функціональні відношення. Оскільки для тексту можлива побудова

декількох коректних структур тексту, що пов’язано з невизначеністю визначення функціональних відношень, необхідним етапом узагальненого алгоритму є пошук оптимальної структури, яка може бути використана для реферату.

Отже, процес автоматизованого реферування тексту на основі розроблених алгоритмів повинен складатися з декількох етапів, основними з яких є аналіз тексту та визначення функціональних відношень, побудова коректних структур тексту на базі цих відношень, пошук оптимальної структури та генерування реферату. Узагальнений алгоритм процесу автоматизованого реферування тексту представлений на рис. 3.1.

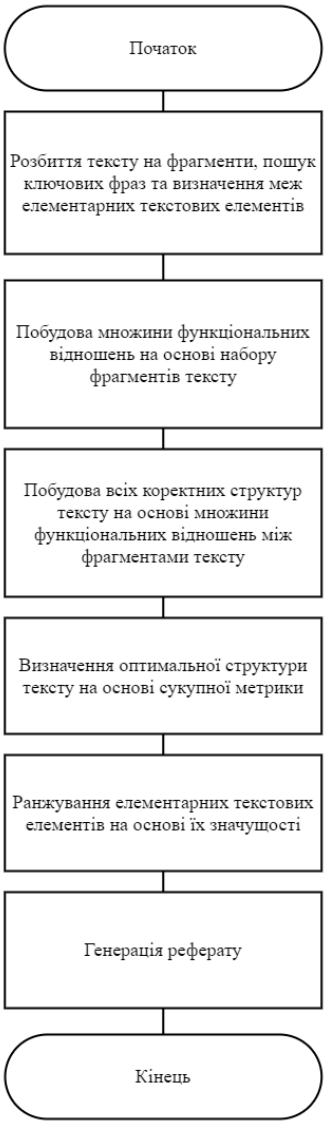


Рис. 3.1. Узагальнений алгоритм автоматизованого реферування

3.2. Алгоритм визначення функціональних відношень між елементами тексту

Згідно з теорією риторичних структур, внутрішня структура текстів характеризується набором функціональних відношень. Однією з головних задач в побудові структури тексту є визначення набору функціональних відношень між елементарними текстовими елементами або частинами речень. Більшість дослідників вважають, що дану проблему можливо вирішити лише використанням глибинного семантичного аналізу текстів, але в даній роботі пропонується вирішувати дану задачу шляхом аналізу ключових фраз [25, 26, 27]. Даних підхід був обраний через проблему відсутності повних баз знань, словників для української мови та спеціалістів в даній області. Для реалізації даного підходу необхідно створити спеціальний словник з ключовими фразами, що буде містити, окрім ключових фраз, додаткову інформацію, таку як функціональні відношення, які з нею асоціюються.

Побудова структури тексту передбачає наступні етапи: визначення функціональних відношень у вхідному тексті та побудова структури на базі визначених функціональних відношень.

Процес побудови функціональних відношень починається з розбиття тексту на елементарні текстові елементи. В якості індикатора між ними можуть використовуватись ключові фрази. В даній роботі пропонується наступна схема їх використання: обираємо список ключових фраз, після чого для кожної з ключових фраз обирається декілька україномовних науково-технічних текстів, аналізується роль даної ключової фрази в тексті та функціональне відношення, що асоціюється з даною фразою. Результатом даної схеми є набір параметрів для кожною ключової фрази, що наведені в таблиці 3.1.

Таблиця 3.1

Параметри ключових фраз

Параметр	Визначення
Орфографічне оточення (Marker)	Включає в себе ключову фразу, знаки пунктуації до та після ключової фрази. Якщо використовується більше однієї ключової фрази, то даний параметр включає в себе їх також
Розташування пов’язаного елементу відносно ключової фрази (Wheretolink)	Включає в себе опис місцезнаходження елементарного текстового елементу, що містить ключову фразу, пов’язану функціональним відношенням з іншим елементарним текстовим елементом. Можливе значення, що може приймати даний параметр: «до» чи «після»
Типи пов’язаних елементів (Typesoftextual units)	Включає в себе опис типів елементів, що пов’язані функціональним відношенням. Можливе значення даного параметру: частина або частини речень, речення або множина речень, параграф або множина параграфів
Відстань між пов’язаними елементами, що вимірюється в елементарних текстових елементах (Clausedistance)	Включає в себе кількість елементарних текстових елементів між елементами, що пов’язані функціональним відношенням.
Відстань між пов’язаними елементами, що вимірюється в реченнях (Sentencedistance)	Включає в себе кількість речень між елементами, що пов’язані функціональним відношенням. Якщо такі елементи знаходяться в одному реченні, даний параметр приймає значення -1.
Відстань до пов’язаного ключового елемента, що є ядром (Distancetosalientunit)	Включає в себе кількість частин речень, що включають в себе ключову фразу, до найбільш значущого елементу-ядра. Є рівним -1, якщо вони знаходяться в одному реченні

Продовження табл. 3.1

Позиція в реченні (Position)	Описує положення ключової фрази в реченні. Можливі значення даного параметр: початок, середина чи кінець речення
Статуси пов’язаних елементів (Statuses)	Включає в себе опис статусів елементів, що пов’язані відношенням. Можливі значення статусів: «ядро» чи «сателіт»
Функціональне відношення (Rhetorical relation)	Включає в себе відношення, яке асоціюється з даною ключовою фразою.
Дія, яку необхідно виконати аналізатору тексту (Breakaction)	Включає в себе одну з наступних інструкцій для аналізатору тексту: nothing, normal, comma, normalThenComma, setAnd, setOr, які призводять до встановлення флагу, що далі буде використовуватись для визначення меж елементарних текстових елементів

Алгоритм визначення функціональних відношень базується на словнику ключових фраз, що отримується на основі аналізу науково-технічних текстів та містить чотири етапи:

- розбиття тексту на речення та визначення для кожного з них набору ключових фраз або дискурсних маркерів;
- визначення меж елементарних текстових елементів;
- визначення функціональних відношень між елементарними текстовими елементами;
- визначення функціональних відношень між елементами, що ще не є пов’язаними.

Розглянемо детальніше кожен з даних етапів.

Перший етап, з якого починається алгоритм визначення функціональних відношень: формування ключових фраз. При цьому спочатку визначаються регулярні вирази для визначення ключових фраз.

Далі аналізується весь текст та запам’ятовується розташування ключових фраз та інших орфографічних маркерів в тексті.

Розглянемо наступний текст в якості прикладу: «Внаслідок віддаленості від Сонця, клімат планети Марс значно жорстокіший за земний. Температура на поверхні зазвичай в середньому досягає -60 градусів Цельсію та може опускатися до -123 градусів Цельсія близько полюсів. Тільки опівдні сонце на тропічних широтах є досить теплим длятанення льоду, але будь-яка вода в рідкому стані випаровується майже миттєво через низький атмосферний тиск.»

Таблиця 3.2

Список регулярних фраз, що асоціюються з ключовими фразами

Маркер	Регулярний вираз
Внаслідок	[\s\t\n]Внаслідок(\s\t\n)
через	[,][s\t\n]через(\s\t\n)
але	,[\s\t\n]але(\s\t\n)
comma	,(\s\t\n)
openParen	[,][\s\t\n]+(
closeParen	)(\s\t\n)
dash	[,][\s\t\n]+-(\s\t\n)
endSetence	(“.”)((“?”)(“!”)(“.””)(“?””)(“!””))
beginParagraph	\s*((\n\t[\s\t]*)|(\n[\s\t\n]{2,}))

Таблиця 3.3

Семантика символів регулярних виразів

Символ	Семантика
\s	пробільний символ
\t	табуляція
\n	перехід на новий рядок
[e]	можливо присутнє
()	групування
a|b	альтернатива
E+	присутнє не менше одного разу
e*	присутнє не менше нуля раз
e{n,}	присутнє принаймні *n* раз

Наступним кроком є розбиття кожного речення на елементарні текстові елементи. На даному етапі аналізатор тексту обробляє текст зліва-направо по кожному реченні та виконує дії, що описані в маркерах. Тобто вхідними даними є речення з пов’язаними з ним набором ключових фраз, а вихідними даними є те ж саме речення, розділене на елементарні текстові елементи, в яких визначені потенціальні ключові фрази, які сигналізують на функціональні відношення. Дія, яку необхідно виконати аналізатору, може приймати значення, наведені в таблиці 3.4.

Таблиця 3.4

Параметри дій аналізатору

Параметр	Визначення
nothing	Аналізатор інтерпретую ключову фразу як звичайне слово. Тобто, границі елементарного текстового елементу не встановлюється
normal	Границя елементарного текстового елементу встановлюється безпосередньо перед ключовою фразою
comma	Границя елементарного текстового елементу встановлюється безпосередньо після першої коми, що знаходиться після ключового слова. Якщо за нею слідує «та» або «або», то границя елементарного текстового елементу встановлюється після наступної коми. Якщо до кінця речення не зустрілось коми, то границя елементарного текстового елементу встановлюється в кінці речення.

Продовження табл. 3.4

normalThenComma	Границя елементарного текстового елементу встановлюється безпосередньо перед ключовою фразою, а друга границя - безпосередньо після першої коми, що знаходиться після ключового слова. Якщо за нею слідує «та» або «або», то границя елементарного текстового елементу встановлюється після наступної коми. Якщо до кінця речення не зустрілось коми, то ця границя встановлюється в кінці речення.
end	Границя елементарного текстового елементу встановлюється безпосередньо після ключової фрази
setAnd	Зберігається інформація, що текст містить лексему «та»
setOr	Зберігається інформація, що текст містить лексему «або»
dual	Встановити границю безпосередньо перед ключовою фразою, якщо нема іншої ключової фрази, що знаходиться безпосередньо перед поточною ключовою фразою. Інакше, аналізатор інтерпретує стан так само як при параметрі *comma*.

Наведемо блок-схему алгоритму на рис. 3.2.

Даний алгоритм обробляє масив ключових фраз зліва-направо на визначає елементарні текстові елементи в реченнях на основі типів ключових фраз, присутніх в кожному реченні. В деякій змінній, назвемо її status, ми будемо зберігати множину ключових фраз, що були оброблені раніше та можуть вплинути на визначення границь елементарних текстових елементів.

Якщо змінна status містить дію *comma*, то поява першої коми, після якої нема ключових слів «та» або «або», визначає границю елементарного текстового елементу.

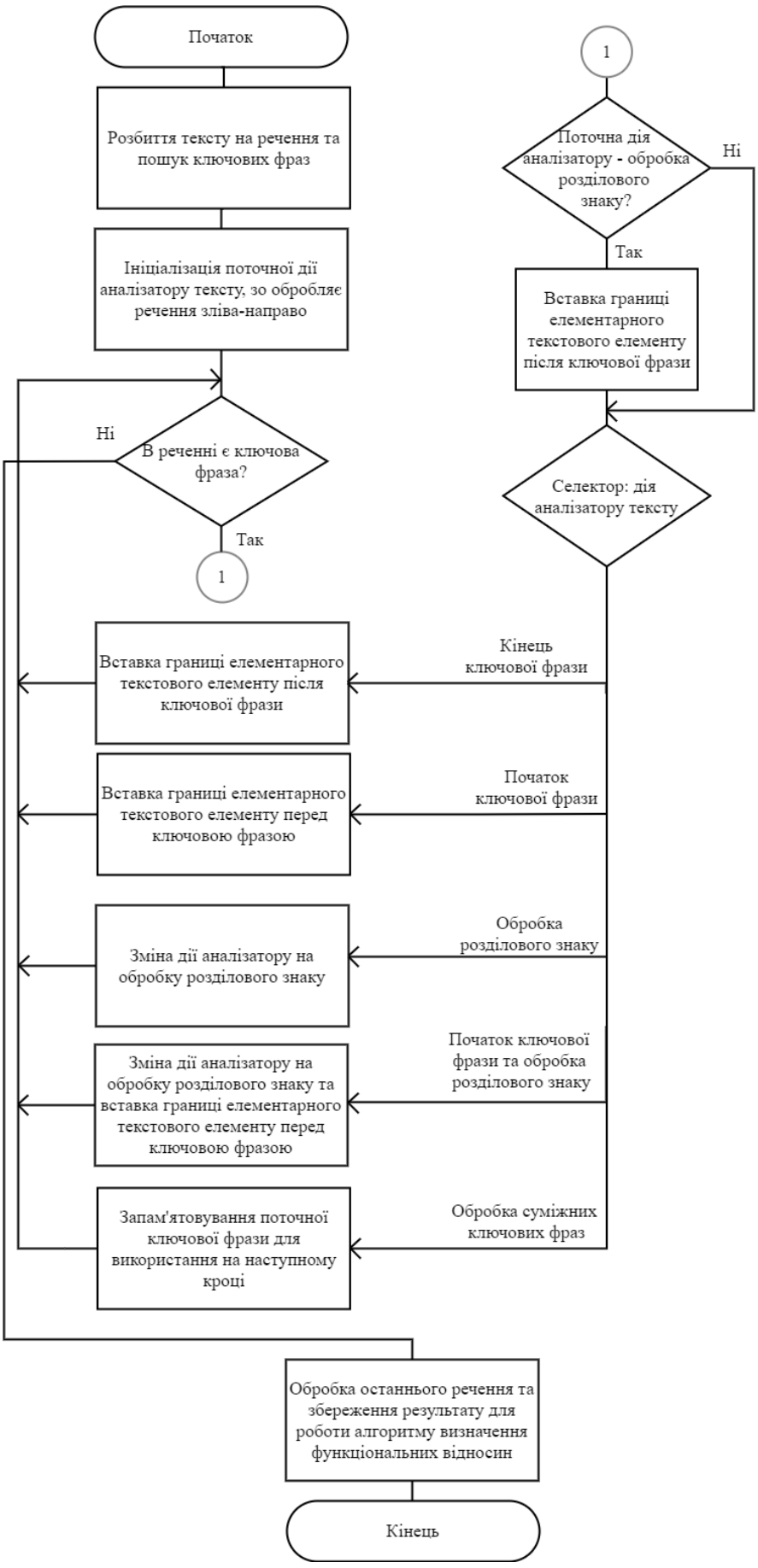


Рис. 3.2. Алгоритм визначення границь елементарних текстових елементів

Розглянемо дії аналізатору, коли змінна *status* не визначена. Якщо тип дії аналізатору є *dual*, то визначення границь залежить від поточної ключової фрази та ключових фраз, що є поточними до неї. Якщо суміжні фрази присутні, то змінна *status* отримує значення *comma*, а інакше визначається границя елементарного текстового елементу. Якщо тип дії аналізатору є normal, то відбувається визначення елементарного текстового елементу перед поточною ключовою фразою. Якщо тип дії аналізатору є comma:якщо поточна ключова фраза є суміжною до попередньої, то попередня також використовується. Інакше змінна *status* оновлюється таким чином, що границя елементарного текстового елементу буде встановлена при першій появі коми. Коли обробляється ключова фраза, типом дії аналізатору якої є *normalThenComma*, алгоритм визначає новий елементарний текстовий елемент, як у випадку типу normal, а далі оновлює status таким чином, що границя буде визначена при першій появі коми. У випадку, коли тип дії аналізатору для ключової фрази є *nothing*, для неї виконується тільки функція присвоєння дискурсної функції.

Після обробки всіх ключових фраз, можливий випадок, що частина тексту залишиться необробленою, в такому випадку буде використана функція, що таким самим чином обробить проміжок тексту від останньої ключової фрази до кінця тексту.

Наступним етапом є розробка процедури визначення функціональних відношень між текстовими елементами різного розміру. На минулому кроці були визначені елементарні текстові елементи та пов’язаний з ними набіл ключових фраз, які є вхідними даними для етапу визначення функціональних відношень. Даний алгоритм ітерує по всім елементарним текстовим елементам та для кожної ключової фрази будує виключну множину відношень, на які сигналізує поточна ключова фраза. Будь-яке відношення з виключної множини відношень може бути використане лише раз. Наприклад, якщо алгоритм знаходиться на і-му елементі послідовності, та вона містить ключову фразу, що сигналізує на функціональне

відношення, що пов’язує поточний елемент з попереднім, що є сателітом.Тоді результатом буде наступний декартовий добуток:

$$\{i, i+1, \dots, i + GetDistanceToSalientUnit(m)\} \times \{i - GetActualDistance(m), i - GetActualDistance(m) + 1, \dots, i-1\}.$$

При цьому $GetDistanceToSailentUnit(m)$ значить отримання значення з поля «*Distancetosalientunit*», а $GetActualDistance(m)$ – отримання поля «*Clausedistance*», якщо елементами відношення є частинами речень або «*Sentencedistance*», якщо вони є реченнями.

Наведений алгоритм автоматично будує виключні гіпотези з множини всіх пар декартового добутку. Вхідними даними є набір елементарних текстових елементів та список відповідних ключових фраз, а вихідними даними є набір виключних гіпотез між елементарними текстовими елементами. Наявність даної проблеми пов’язано з тем, що не завжди ключова фраза однозначно сигналізує на одне функціональне відношення.

Наведемо блок схему даного алгоритму на рис. 3.3.

Слід зазначити, що не завжди алгоритм визначення функціональних відношень за допомогою ключових фраз визначає відношення для всіх частин речення. Для елементів, що не пов’язані функціональним відношенням, необхідно виконати додаткові дії для визначення відношень між ними. Згідно з даним алгоритмом, якщо два речення описують одне й те саме, то вірогідно, що друге речення деталізує попереднє. Інакше вони відносяться до різних тем. Висновок згідно їх подібності визначається через кількість схожих слів у реченнях: якщо воно більше за порогове значення, то встановлюється функціональне відношення «уточнення», інакше «з’єднання». Вхідними даними даного алгоритму є набір елементарних текстових елементів та визначені на минулому кроці набір функціональних відношень. Вихідними даними є повний набір функціональних відношень між елементарними текстовими елементами.

Блок-схема даного алгоритму зображена на рис. 3.5.

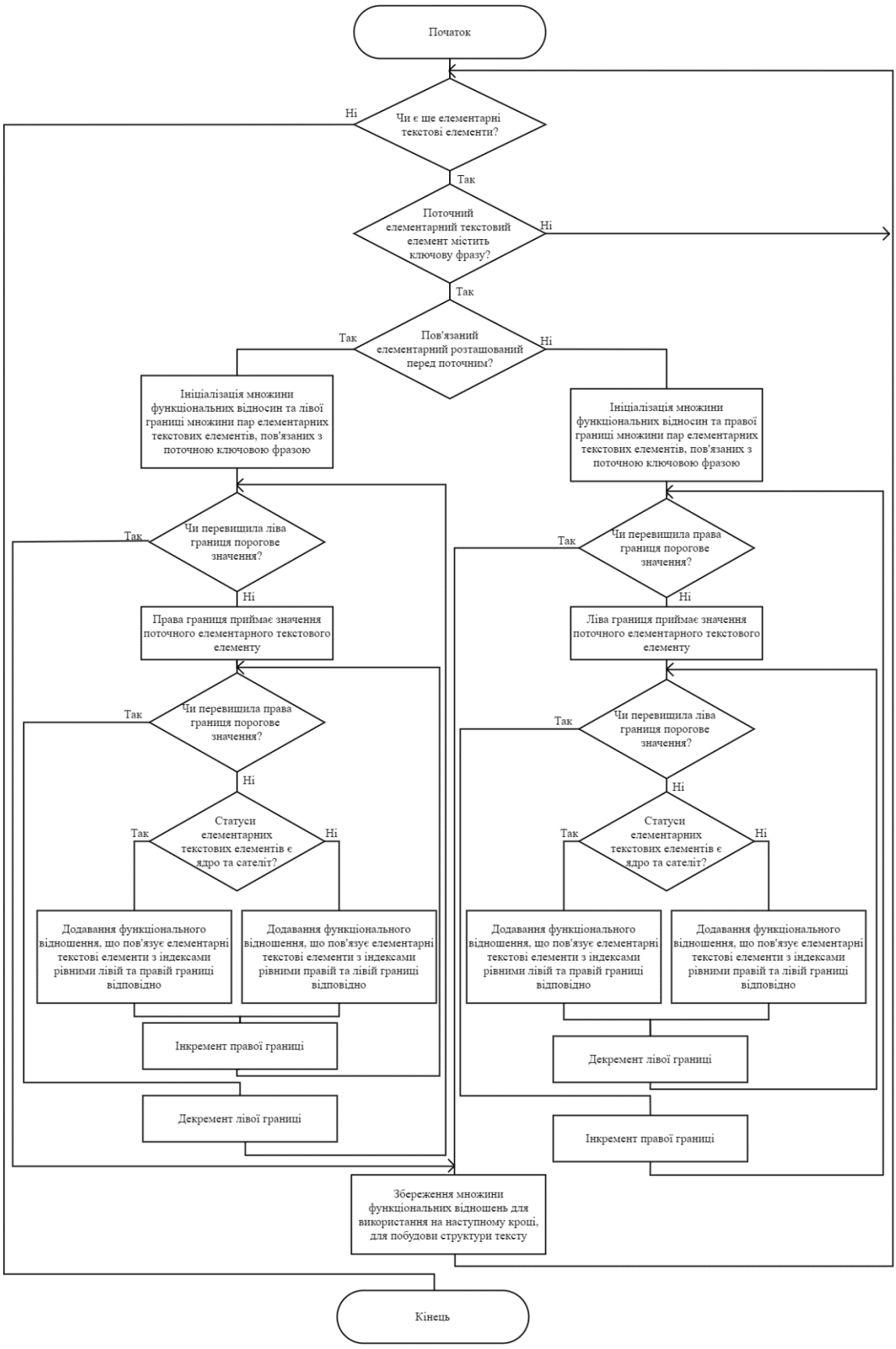


Рис. 3.3. Алгоритм побудови множини функціональних відношень на основі набору елементарних текстових елементів

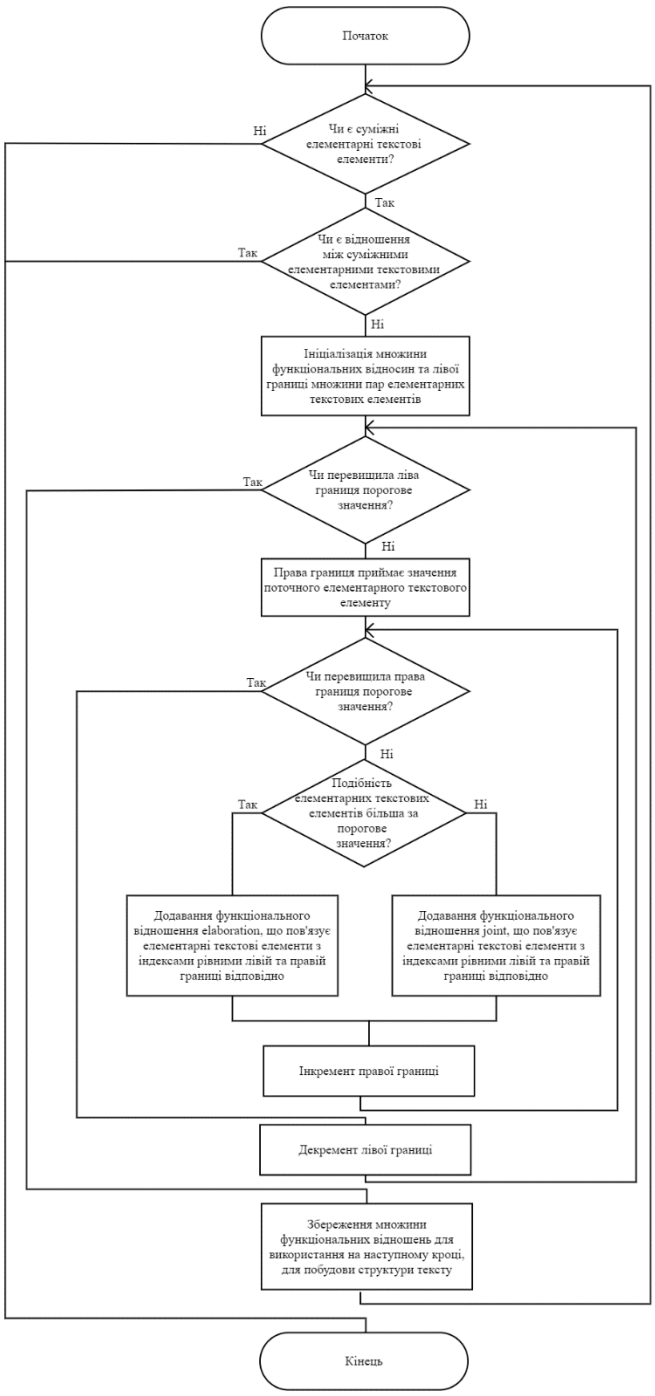


Рис. 3.5. Алгоритм визначення функціональних відношень для непов’язаних елементарних текстових елементів

В ході аналізу було визначено, що якщо кількість речень в параграфі невелике та вони не містять ні одної ключової фрази, то між ними міститься функціональне відношення «уточнення».

3.3. Алгоритм побудови структури тексту

Алгоритм побудови коректної структури тексту приймає в якості вхідних даних множину функціональних відношень між фрагментами тексту та визначає структуру тексту. Передбачається, що текст має ієрархічну структуру, що виражається в наявності ключових фраз між реченнями та параграфами. Дане припущення пов'язано з підвищенням ефективності роботи алгоритму, оскільки при відсутності чітких меж між реченнями та параграфами кількість можливих варіантів побудови структури тексту значно підвищується.

Задача побудови структури тексту може бути сформульована наступним чином: є послідовність елементарних текстових елементів $U = u_1 u_2 \dots u_n$ та множина функціональних відношень RR, що лежать між елементарними текстовими елементами множини U, з якої необхідно знайти всі коректні структури тексту.

Даний підхід передбачає побудову дерева знизу догори.

Ідея алгоритму, що базується на даному підході, полягає в наступному: на початку роботи кожен i-ий елементарний текстовий елемент асоціюється з елементарним деревом, тобто деревом, що складається лише з одного елементу, що має статус ядра чи сателіту та множину найбільш важливих елементарних текстових елементів, що є нащадками $\{i\}$. Спочатку будь-яке відношення з множини RR може бути використано для зв'язку двох елементів в більш складні дерева. Після побудови всіх елементарних дерев структура тексту може бути отримана шляхом зв'язку суміжних дерев у більші, при умові що на кожному кроці отримується коректна деревовидна структура. З кожним кроком пов'язана множина функціональних відношень, які можуть бути використані на наступних кроках. На початку елементарне дерево може бути перетворено в більше дерево, використовуючи весь набір RR, але як тільки одне з відношень було використано, воно стає недоступним для подальшого використання.

Параметрами алгоритма побудови структури тексту є:

- множина елементарних текстових елементів $U = u_1 u_2 \dots u_n$;
- множина констант «*ядро*», «*сателіт*», «*листок*», «*null*»;
- назви всіх функціональних відношень;
- об'єкти типу $tree(status, type, valuable, left, right)$.

Об'єкти типу $tree(status, type, valuable, left, right)$ забезпечують функціональне представлення коректних дерев. Змінна $status$ може приймати значення «*ядро*» чи «*сателіт*»; $type$ містить в собі назву функціонального відношення; $valuable$ містить в собі підмножину елементів з множини елементарних текстових елементів; $left$ та $right$ можуть приймати значення або null, або бути рекурсивним визначенням через об'єкти типу $tree$.

Об'єкти типу $tree(status, type, valuable, left, right)$ відповідають коректній текстовій структурі тільки в тому випадку, коли аргументи $status, type, valuable$ мають такі самі значення, як у кореня текстової структури та якщо $left$ та $right$ відповідають лівому та правому піддеревам відповідно.

Для розробки алгоритму необхідно ввести наступні предикати:

- $isFunctionalRelPresent(rr)$ є істинним для тексту тільки тоді, коли функціональні відношення в rr лежать між елементарними текстовими елементами в тексті;
- $S(l, h, tree(\dots), R_{lh})$ є істинним, коли коректна текстова структура, що відповідає аргументу $tree(\dots)$, може бути побудована на текстовому відрізку $[l, h]$, використовуючи функціональні відношення між елементами цієї частини тексту. Аргумент R_{lh} містить множину функціональних відношень, що можуть бути використані для розширення дерева $[l, h]$, тобто відношення, що лежать між елементами тексту та ще не були використані в побудові дерева $tree(\dots)$;

- $isAsymmetric(name)$ є істинним, якщо $name$ – асиметричне відношення;
- $isSymmetric(name)$ є істинним, якщо $name$ – симетричне відношення.

Тепер визначимо множину правил виводу, що будемо використовувати при побудові коректної структури тексту. Для початку, слід привести вже відомі два предикати, що використовуються при визначенні правил виводу:

$$isFunctionalRelPresent(RR) \tag{3.1}$$

$$isValidPosition(u_i, j) \tag{3.2}$$

$$[isValidPosition(u_i, j) \cap isFunctionalRelPresent(RR) \rightarrow S(j, j, tree(\text{сателіт}, \text{листок}, \{u_i\}, null, null), RR)] \tag{3.3}$$

$$[isValidPosition(u_i, j) \cap isFunctionalRelPresent(RR) \rightarrow S(j, j, tree(\text{ядро}, \text{листок}, \{u_i\}, null, null), RR)] \tag{3.4}$$

Таким чином, текст з N елементарних текстових елементів може мати N правил виводу, що використовують форму (3.3) та N правил, що використовують форму (3.4).

Оскільки набір RR може містити взаємовиключні відношення, необхідно бути впевненим, що відношення з таких множин використовується лише раз.

Припустимо, що існує два текстових елементи: $[l, b]$, який характеризується структурою $tree_1(\dots)$ та функціональними відношеннями rr_1 та другий: $[b+1, h]$, який характеризується структурою $tree_2(\dots)$ та функціональними відношеннями rr_2. Також, припустимо що функціональне відношення $isInFunctionalRel(name, s, n)$ лежить між елементом s, що міститься в множині valuableелементу $[l, b]$ та елементом n, що міститься в множині *valuable* елементу $[b+1, h]$ та відношення $isInFunctionalRel(name, s, n)$, що може бути використано для поєднання $[l, b]$ та $[b+1, h]$ є асиметричним. В такому випадку, елементи $[l, b]$ та $[b+1, h]$ можуть бути поєднані в текстовий елемент $[l, h]$ з статусом

«ядро» чи «сателіт», типом name та множина valuable першого елементу є рівною відповідній множині другого елементу. Множина функціональних відношень, що можуть бути використані для подальших перетворень, являє собою $rr_1 \cap rr_2 \backslash_{\oplus} \{isInFunctionalRel(name, s, n)\}$, де оператор $\backslash_{\oplus}$ значить, що використовувати відношення з множини можна лише раз. Аналогічно описуються випадки для симетричних та розширених відношень.

Опишемо алгоритм побудови всіх можливих коректних дерев. Вхідними даними є текст T з nелементарних текстових елементів та множина RR функціональних відношень між елементарними текстовими елементами. Вихідними даними є всі коректні дерева.

Спочатку для вхідної множини функціональних відношень RR між елементарними текстовими елементами застосовується правило (3.1). Далі у випадку успіху для кожного з множини елементарних текстових елементів застосовуємо правила (3.2) - (3.4), які дадуть нам листки майбутнього дерева. Далі використовуємо критерій визначення коректного дерева для всіх пов'язаних функціональним відношенням текстових елементів розміром від 1 до n-1. В результаті ми отримаємо набір результуючих правил виводу з $S(l, h, tree(\dots), RR)$, де l та h є першим та останнім елементарним текстовим елементами, отже даний набір покриває увесь текст.

Процес побудови тексту не є однозначним, тому що для одного тексту можна побудувати декілька структур. Розглянемо наступне речення: [Тільки опівдні сонце на тропічних широтах є досить теплим длятанення льоду,]^1 [але будь-яка вода в рідкому стані випаровується майже миттєво]^2[через низький атмосферний тиск.]^3.

Для нього можна побудувати чотири дерева, зображених на рис. 3.6.

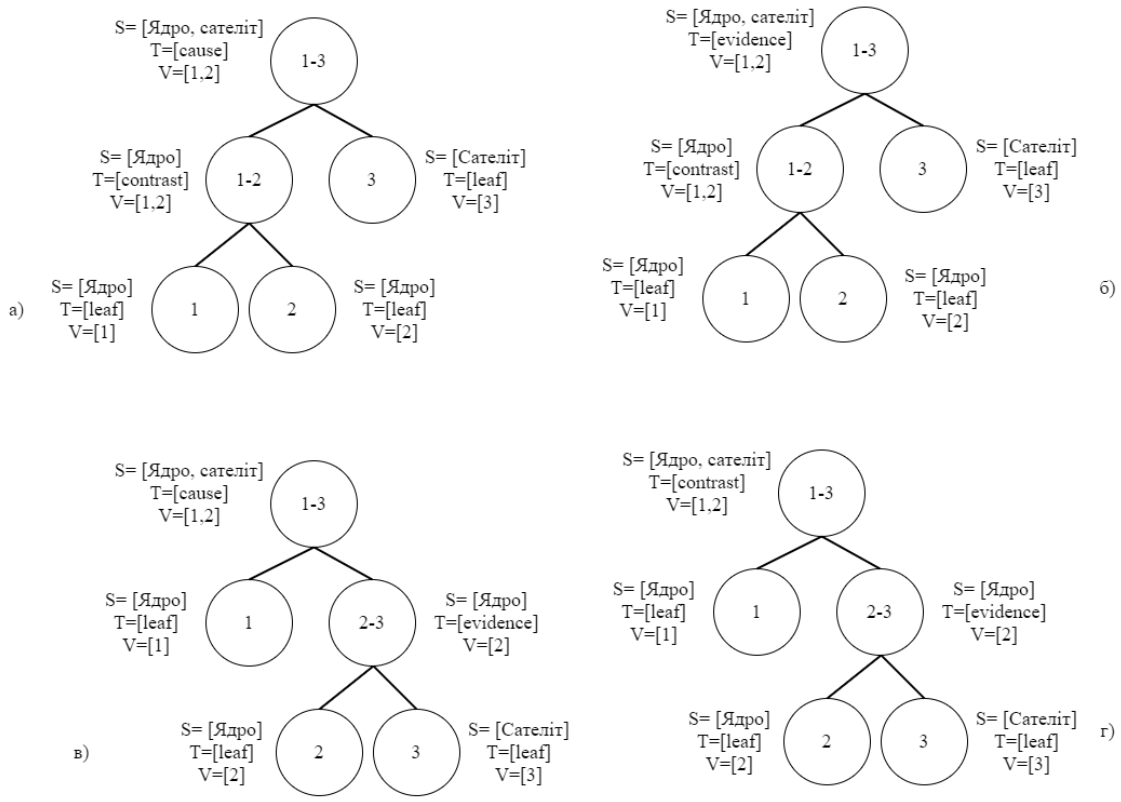


Рис. 3.6. Приклади дерев

В даній роботі для вибору коректної структури тексту пропонується використовувати комбінацію індикаторів, що описані нижче.

Кластерний індикатор: даний індикатор передбачає, що оптимальною є структура тексту, що відображає найбільшу кількість тематичних границь тексту, на основі якого вона побудована [28]. Для цього з кожним вузлом дерева пов’язується його кластерна вага: для листків дана вага дорівнює нулю, а для вузлів він обчислюється на основі схожості його потомків. Отже, кластерний індикатор для дерева можна порахувати на основі кластерних ваг його вузлів.

Ключові фрази: даний індикатор передбачає, що кращим є те дерево, що використовує більше функціональних відношень.

Заголовний: даний індикатор обраховується для структури тексту на основі схожості заголовку та найбільш важливих елементарних текстових елементів. Він передбачає, що більш важливим є те дерево, у якого заголовна вага є більшою.

Позиційний: даний індикатор базується на позиції важливих речень в тексті та обраховується шляхом призначення дадатньої ваги кожному текстовому фрагменту, що знаходиться в перших або останніх двох реченнях параграфу. Для структури тексту даний індикатор обраховується як середнє значення між позиційними вагами найбільш важливих вузлів структури. Він передбачає, що кращим є те дерево, позиційна вага якого більша.

Формовий: при аналізі текстів було виявлено, що найбільш оптимальним є дерева, які перехилені на праву сторону. Даний факт може бути пояснений тим, що обробка тексту відбувається зліва-направо та у більшості випадків найбільш важлива ідея описана на початку тексту, яка в подальшому розкривається, доповнюючи та додаючи деталей до основної ідеї тексту.

Для того, щоб зробити процес побудови структури тексту однозначним, наведемо рекурентну формулу для обчислення ваги структури тексту з метою вибору найбільш оптимальної структури тексту. Дана вага обраховується рекурсивно як сума ваг та різниць висот гілок дерева.

$$w(tree) = \begin{cases} 0, \text{якщо } isLeaf(tree) \\ w\big(leftOf(tree)\big) + w(rightOf(tree)) + \\ depth(rightOf(tree)) - depth\big(leftOf(tree)\big), \text{інакше} \end{cases} \quad (3.5)$$

Таким чином, загальний індикатор для визначення оптимального дерева обраховується на основі лінійної комбінації всіх індикаторів:

$$s = w_{clust} \times s_{clust} + w_{mark} \times s_{mark} + w_{shape} \times s_{shape} + w_{title} \times s_{title} + w_{pos} \times s_{pos} \quad (3.6)$$

Конкретні значення ваг індикаторів знаходяться в інтервалі від 0 до 1 та визначаються експериментальним методом.

3.4. Алгоритм отримання реферату

Як було встановлено, елементарні текстові елементи найважливіших нащадків дерева є найбільш значущими частинами вузлів. Ця інформація може бути використана для створення реферату з вхідного тексту. Маючи вказану множину для кожного вузла, ми можемо отримати відсортований по значущості список елементарних текстових елементів. Визначення значущості елементарного текстового елементу базується на припущенні, що елементарні текстові елементи, що містяться в множині нащадків верхніх вузлів є більш значущими за елементарні текстові елементи, що містяться в множині нащадків нижніх вузлів.

Найбільш простим способом визначення значущості елементарного текстового елементу – це підрахунок ваги кожного елементарного текстового елементу на основі аналізу висоти дерева відносно того вузла, де вперше зустрівся даний елементарний текстовий елемент в множині ключових нащадків. Значущість елементарного текстового елементу визначається його вагою.

Маючи даний відсортований по значущості елементарних текстових елементів список, ми можемо отримувати реферати різного обсягу. Наприклад, якщо ми хочемо отримати дуже стислий реферат, ми можемо згенерувати текст лише з одного елементарного текстового елементу.

Алгоритм отримання реферату для тексту:

Вхідні дані: текст *T*, число *p* що є обсягом реферату у відсотках, від 1 до 100.

Вихідні дані: набір *p*% найбільш значущих елементарних текстових елементів.

1. Визначити структуру тексту за допомогою алгоритму побудови структури тексту.
2. Відсортувати список елементарних текстових елементів за їх значущістю.

3. Обрати необхідний відсоток елементарних текстових елементів з початку даного відсортованого списку для створення реферату.

Таким чином, розроблені алгоритми дозволяють автоматично побудувати структуру тексту, після чого, на основі відсортованого за значущістю списку елементарних текстових елементів, отримувати реферат згідно вказаного обсягу.

3.5. Висновки до третього розділу

У даному розділі розроблений вузькоспеціалізований словник ключових фраз, який за рахунок збереження додаткової інформації враховує специфіку функціональних відношень між фрагментами тексту, що дозволяє визначати множину функціональних відношень для україномовних науково-технічних текстів.

Також, розроблений алгоритм визначення функціональних відношень між фрагментами тексту, що базується на використанні словника ключових фраз та процедурами аналізу функціональних відношень між ними, що дозволяє зменшити інформаційну надлишковість за рахунок відмови від використання словників загального призначення.

Розроблений алгоритм побудови структури тексту на базі множини функціональних відношень між фрагментами тексту, що враховує проблему неоднозначності визначення функціональних відношень за допомогою ключових фраз шляхом генерації множини альтернативних варіантів коректних структур тексту та відбору з них оптимального за критерієм сукупної метрики.

Розроблений алгоритм отримання реферату з вхідного тексту шляхом відбору необхідної кількості елементарних текстових елементів з заздалегідь відсортованого за значущістю масиву елементарних текстових елементів.

4. ПОБУДОВА СИСТЕМИ АВТОМАТИЗОВАНОГО РЕФЕРУВАННЯ ТЕКСТУ ТА АНАЛІЗ ЕФЕКТИВНОСТІ РОЗРОБЛЕНОГО МЕТОДУ

4.1. Загальна структура організації системи автоматизованого реферування тексту

Запропонована технологія створення програмної системи, що реалізує даний метод, базується на використанні загальних принципів системного проектування [29] та об'єктно-орієнтованого програмування [30].

Структура системи, що реалізує вказаний метод, зображена на рис. 4.1.

Взаємозв'язок підсистем забезпечує керуюча програма. Процес починається з передачі управління підсистемі аналізу тексту, яка за допомогою словника ключових фраз виконує процес розбиття тексту на частини, такі як параграфи, речення та частини речень та визначає функціональні відношення між ними. Далі управління передається підсистемі визначення коректної структури тексту, яка, використовуючи результати минулого етапу, будує результуюче дерево, що покриває увесь текст. На фінальному етапі працює підсистема отримання реферату з отриманої коректної структури, основною задачею якої є ранжування елементарних текстових елементів з результуючого дерева за їх значущістю та створити реферат згідно вказаних параметрів.

Розглянемо роботу кожної підсистеми окремо.

Підсистема аналізу тексту приймає в якості вхідних даних текст, з якого необхідно отримати реферат. Головними задачами даної підсистеми є:

- розбиття тексту на частини;
- визначення функціональних відношень між ними.

Словник ключових фраз та параметри, що пов'язані з кожною з них міститься у текстовому файлі, з яким взаємодіє підсистема аналізу тексту. Додаткові параметри, які пов'язані з ключовими фразами та є необхідними

для визначення функціональних відношень, такі як ідентифікатор ключової фрази, відношення, з яким воно асоціюється, регулярний вираз та дія аналізатору тексту.

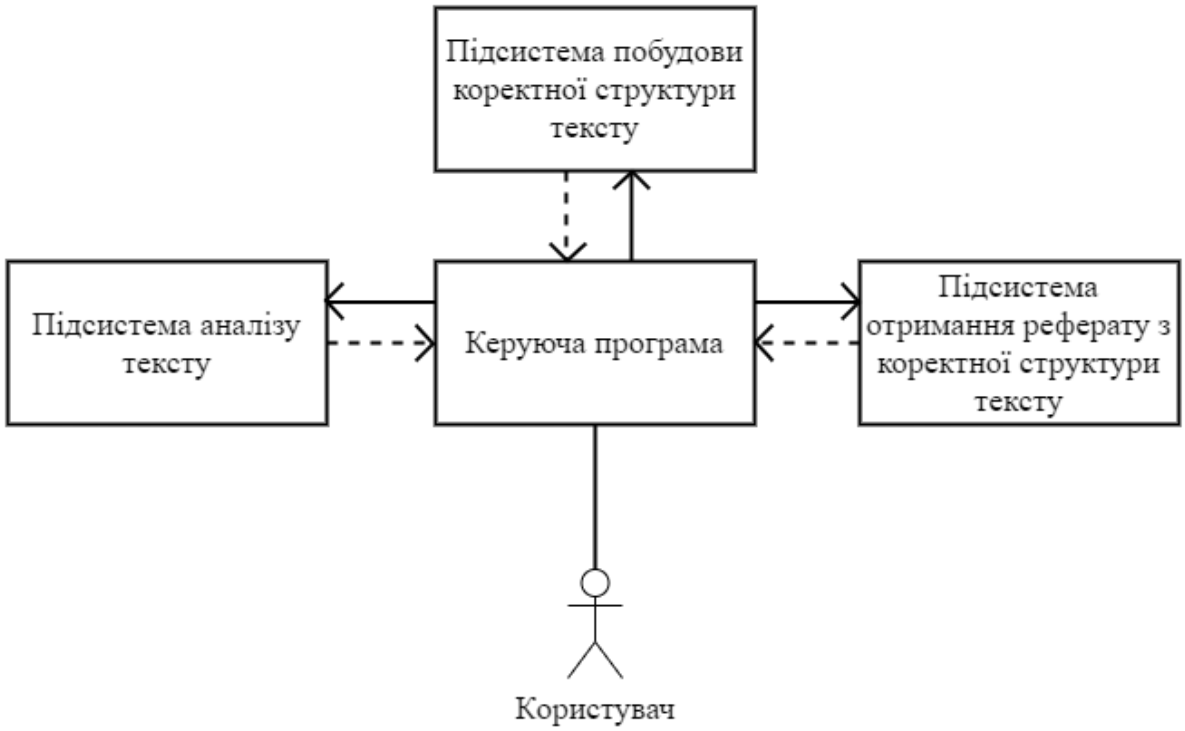


Рис. 4.1. Структура системи автоматизованого реферування

Також, при визначенні границь елементарних елементів, заповнюється масив додаткових параметрів для визначення функціональних відношень, що містить наступні поля:

- ідентифікатор ключової фрази;
- статуси елементарних текстових елементів, що поєднує дана ключова фраза (*Statuses*);
- типи елементарних текстових елементів, що поєднує дана фраза (*Types*);
- місцеположення другого елементарного текстового елементу, з яким пов’язаний поточний елементарний текстовий елемент, що містить ключову фразу (*WhereToLink*);

- відстань між елементарними текстовими елементами, що пов’язані функціональним відношенням , що вимірюється в елементарних текстових елементах(*ClauseDistance*);
- відстань між елементарними текстовими елементами, що пов’язані функціональним відношенням , що вимірюється в реченнях (*SentenceDistance*);
- відстань до елементарного текстового елементу, що є ядром у відношенні між двох елементарних текстових елементів, що поєднує дана ключова фраза (*DistanceToSalientUnit*).

Дані параметри використовуються при визначенні границь для елементарних текстових елементів та визначенні функціональних відношень між ними.

Дана підсистема повертає набір елементарних текстових елементів та функціональних відношень між ними, що є вхідними даними для підсистеми побудови коректної структури тексту. Розглянемо роботу даної підсистеми детальніше. Для отримання результуючого дерева підсистема виконує наступні дії:

- визначаємо листки дерева за допомогою правил (3.1) – (3.4) та шукаємо елементи для поєднання, виключаючи використані функціональні відношення з множини доступних для використання;
- пошук оптимального дерева для кожного з рівнів, згідно з загальним індикатором, що обраховується на основі лінійної комбінації всіх індикаторів;
- об’єднання оптимальних дерев всіх рівнів в одне, що покриває увесь текст.

Підсистема отримання анотації з коректної структури тексту приймає в якості вхідних даних приймає дерево, що покриває увесь текст а також параметр, що визначає розмір анотації. Вихідними даними даної підсистеми є текст, що складається з елементарних текстових елементів, відсортованих за значущістю.

Основними діями даної підсистеми є:

- ранжування листків результуючого дерева за їх значущістю на основі критерію їх положення в множинах елементарних текстових елементів у вузлах;
- вибір необхідної кількості елементарних текстових елементів для створення реферату.

4.2. Обґрунтування вибору засобів розроблення

Програмна реалізація системи автоматизованого реферування розроблена з використанням мови програмування Java. В якості середовища розроблення була використана IntelliJIDEA 2020.1.2.

Коротко розглянемо обрані засоби.

4.2.1. *Мова програмування Java*

Java є об’єктно-орієнтованою мовою програмування, що була випущеною компанією «SunMicrosystems»в 1995 році, але в 2009 році компанія «Oracle» придбала компанію «Sun Microsystems»та почала займатися мовою програмування Java [31]. Однією з головних переваг цієї мови є її незалежність від платформи, на якій буде запускатись програмне забезпечення, написано на даній мові. Всі сутності в мові програмування Java є об’єктами, за виключенням базових типів, такі як числа.

Дану мову обрано для використання при розробці системи автоматизованого реферування тексту через наступні переваги:

- незалежність від платформи;
- автоматичне керування пам’яттю;
- об’єктно-орієнтованість.

4.2.2. *IntelliJ IDEA 2020.1.2*

IntelliJ IDEA – продукт компанії JetBrains [32]. Вперше версія з’явилась в 2001 році та швидко стала популярною через велику кількість

вбудованих інструментів, такі як інструменти для рефакторингу коду, чого не було в інших середовищах розробки для мови програмування Java. До того ж, дане середовище розробки підтримує багато плагінів, що допомагають у створенні покриття коду тестами та зручної роботи з системами конролю версій.

4.3. Програмна реалізація методу автоматизованого реферування україномовного науково-технічного тексту

Розглянемо основні класи підсистеми аналізу тексту в табл. 4.1.

Таблиця 4.1

Опис основних класів підсистеми аналізу тексту

Назва класу	Опис
Parser	Розбиває весь текст на параграфи, речення та частини речень, використовуючи набір регулярних виразів. Також знаходить ключові фрази.
TextualUnitSegmentationalAlgorithm	Розбиває кожне речення на частини.
RelationHypothesisAlgorithm	Визначає для кожного рівня текстових елементів функціональні відношення, базуючись на знайдених ключових фразах.
WordConectBasedHypothesisAlgorithm	Визначає функціональні відношення для кожного рівня текстових елементів, базуючись на зв’язності цих елементів. Використовується тільки для тих текстових елементів, для яких не вдалося знайти функціональні відношення на минулому кроці.

Продовження табл. 4.1

ElementaryUnit	Елементарний текстовий елемент
Relation	Функціональне відношення
TextElement	Базовий клас для фрагменту тексту, нащадками якого є глава, параграф та речення.
Section	Глава
Paragraph	Параграф
Sentence	Речення
DiscourseMarker	Ключова фраза

Представимо дані класи на діаграмі класів на рис. 4.2.

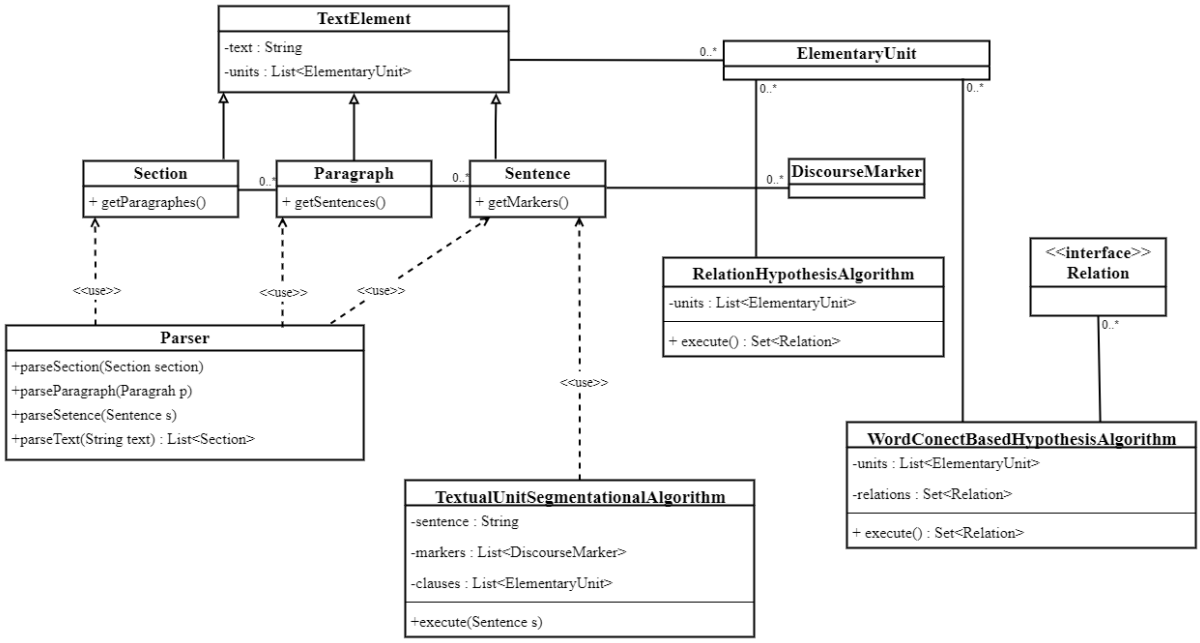


Рис. 4.2. Діаграма класів підсистеми аналізу тексту

Основні класи підсистеми побудови коректної структури тексту представлені в табл. 4.2.

Таблиця 4.2

Опис основних класів підсистеми побудови коректної структури тексту

Назва класу	Опис
ProofTheoryAlgorithm	Шукає всі коректні структури для даного набору текстових елементів

Продовження табл. 4.2

StandartValidTreeFinder	Визначає оптимальну структуру, базуючись на множині коректних структур
Relation	Функціональне відношення
Axiom	Інтерфейс правила виводу
ElementaryUnit	Елементарний текстовий елемент
ValidTreeFinder	Інтерфейс пошуку оптимального дерева
TreeAxiom	Правило виводу типу $tree(\dots)$

Діаграма основних класів підсистеми побудови коректних структур тексту зображена на рис. 4.3.

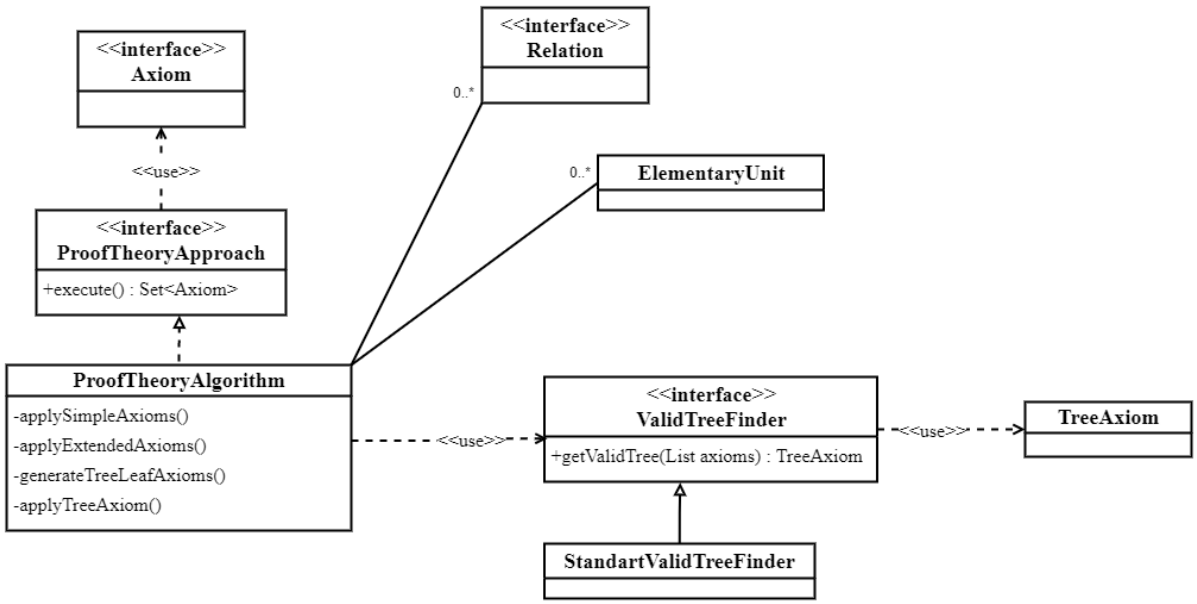


Рис. 4.3. Діаграма класів підсистеми побудови коректних структур тексту

Основні класи, що описують правила виводу, що використовуються для побудови структури тексту, представлені в табл. 4.3

Таблиця 4.3

Опис основних класів, що описують правила виводу для побудови структури тексту

Назва класу	Опис
Axiom	Інтерфейс правила виводу

TreeAxiom	Правило виводу типу $tree(\dots)$
Relation	Інтерфейс функціонального відношення
BaseTreeRelation	Базовий клас для правил виводу
BaseSimpleAxiom	Базовий клас для правил виводу, що містять просте функціональне відношення
BaseExtendedAxiom	Базовий клас для правил виводу, що містять розширене функціональне відношення

Діаграма класів цих правил виводу представлена на рис. 4.4.

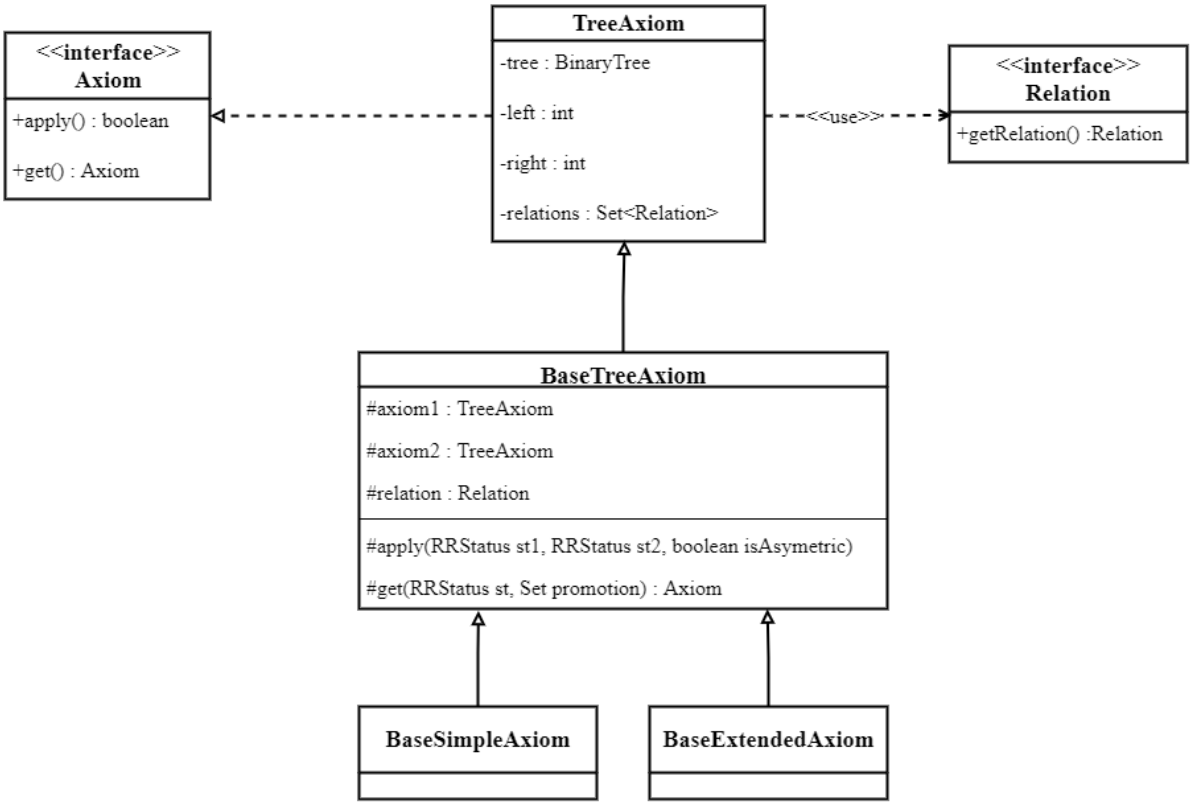


Рис. 4.4. Діаграма класів правил виводу, що використовуються в алгоритмі побудови структури тексту

4.4. Оцінювання ефективності методу автоматизованого реферування україномовного науково-технічного тексту

Оцінювання ефективності розроблених методів та алгоритмів проводилось за принципом оцінки функціональної ефективності.

Функціональна ефективність розробленого методу та алгоритмів автоматизованого реферування тексту оцінюється по якості рефератів, отриманих в результаті роботи програми. Однак, об’єктивна оцінка якості автоматизованого реферування тексту є непростою задачею. Саме поняття реферування заздалегідь передбачає суб’єктивний характер оцінки результатів.

Аналіз робіт з автоматизованого реферування показав, що на сьогоднішній день методи оцінки рефератів базуються лише на експертних оцінках. Велика кількість недостатньо формалізованих параметрів, такі як «релевантність» або «ступінь задоволення користувача», навіть для професіоналів значно ускладнює задачу оцінки переваг та недоліків застосування різних методів при вирішенні актуальних задач.

Існує два основних метода оцінки якості автоматично згенерованого реферату: «зсередини» та «ззовні». Метод «ззовні» передбачає оцінку реферату по критерію, наскільки даний реферат допоміг в вирішенні конкретної проблеми, таких як: знайти певну інформацію або відповісти на питання по початковому тексту і т.д.

Метод «зсередини» базується на суб’єктивній оцінці реферата користувачем по різним критеріям. Наприклад, наскільки повно реферат відображає зміст первинного документу, чи є надлишковість в рефераті, або порівнюють даний реферат з «еталоном», тобто написаним автором або експертами в даній області.

Процедури оцінки результатів відрізняються, але серед них можна виділити наступні спільні риси:

- Рівноправність систем. Процедура оцінки повинна, за можливістю, гарантувати рівноправність методів та алгоритмів при оцінці результатів.
- Анонімність джерела результату. При проведенні оцінки результатів роботи повинна зберегтись анонімність джерела результату, тобто

ті, хто оцінює результати роботи, не повинні знати, який саме алгоритм видав той чи інший результат.

Оцінка ефективності рефератів, отриманих за допомогою розроблених методів та алгоритмівпроводилась на основі залучення експертної групи.

Методи експертних оцінок – це методи організації роботи з експертами та обробки їх точок зору. Якість отриманих експертних оцінок в значній мірі визначається підготовкою експертизи та застосовуваними методами обробки інформації, отриманої від експертів. Єдиних правил підготовки та проведення експертної оцінки нема.

Однак, можна виділити основні етапи її підготовки та проведення. До них відносяться:

- формулювання мети експертного аналізу;
- розробка процедури проведення експертної оцінки;
- підбір експертів;
- обробка результату опитування та аналіз отриманих даних.

Розглянемо детальніше окремі стадії експертного оцінювання.

Мета експертного аналізу – визначити ефективність розроблених методів та алгоритмів формування рефератів. Це можливо зробити на основі оцінки якості згенерованих рефератів.

Процедура проведення експертної оцінки при виборі значущих фрагментів тексту полягала в наступному.

Кожному експерту було видано 5 одних і тих самих науково-технічних текстів обсягом від 150 до 500 слів.

Тексти відрізнялись між собою не тільки за обсягом, а й за кількістю ключових фраз в тексті. Це пов’язано з тим, що в розробленому методі важливі фрагменти тексту обираються на основі побудованої структури тексту, визначеної відповідно з множиною функціональних відношень. Оскільки функціональні відношення визначаються на основі ключових

фраз, то найбільш значущим параметром для даного методу є кількість ключових фраз.

Далі був визначений мінімальний набір текстових елементів для кожного тексту. Кожен елемент був розділений квадратними дужками та мав порядковий номер від 1 до N, де N – кількість елементарних текстових елементів в тексті.

Сенс методу оцінювання елементарних текстових елементів полягає в наступному: експерт ставить кожному елементарному елементу свою оцінку його значущості . Метод дозволяє давати одну й ту саму оцінку декільком елементарним текстовим елементам. Саме такий метод оцінювання використовується на іспитах. В даному випадку використовується наступна шкала балів: 0 балів, якщо елементарний текстовий елемент не є значущим; 1 бал, якщо елементарний текстовий елемент є відносно значущим та в 2 бали оцінюються найбільш значущі елементарні текстові елементи.

Результати даної оцінки представлені в табл. 4.3.

Таблиця 4.3

Результати оцінки розробленого алгоритму на основі експертних оцінок

Елементарний текстовий елемент	Експерти					Система
	1	2	3	4	5	
1	0	1	0	1	0	3
2	2	2	2	2	2	6
3	1	1	1	2	1	4
4	0	1	0	0	0	3
5	1	1	0	0	0	3
6	0	0	0	0	0	1
7	0	1	1	0	0	3
8	2	2	2	2	2	5
9	0	1	1	0	0	3
10	0	1	1	1	1	4

Дана таблиця містить оцінки експертів відносно кожного елементарного текстового елементу в тексті та ваги, що була отримана з результуючої структури тексту, отриманої за допомогою алгоритму.

На етапі обробки результатів опитування експертів формується узагальнена оцінка опитування всієї групи експертів. Зрозуміло, що точки зору експертів відрізняються між собою, але необхідно зрозуміти наскільки сильно відрізняються їх оцінки. Якщо відхилення є малим – доцільно буде усереднити їх оцінки, щоб виключити випадкові відхилення в той чи інший бік. Таким чином, групова оцінка експертних точок зору може вважатися достатньо надійною тільки при умові узгодженості відповідей кожного із спеціалістів. Інакше, якщо відхилення є великим, то усереднення результатів є формальною процедурою.

Оцінка узгодженості відповідей експертів розраховується на основі коефіцієнту узгодженості Кендалла [32] за наступною формулою:

$$W = \frac{12S}{n^2(m^3 - m)},$$

де S – сума квадратів відхилень всіх оцінок рангів кожного об'єкта експертизи від середнього значення; n – число експертів та m – число об'єктів експертизи. Значення коефіцієнту узгодженості варіюється від 0 до 1, де 0 – повна неузгодженість та 1 – повна узгодженість.

Результати узгодженості оцінок експертів представлені в табл. 4.4.

Таблиця 4.4

Результати узгодженості експертних оцінок

	Текст 1	Текст 2	Текст 3	Текст 4	Текст 5	Підсумок
Всі елементарні текстові елементи	0.7264	0.7323	0.6923	0.6889	0.7008	0.7067

Продовження табл. 4.4

Найбільш важливі елементарні текстові елементи	0.8846	0.6307	0.6483	0.6373	0.6730	0.6566
Елементарні текстові елементи середньої значущості	0.5128	0.7307	0.5384	0.4615	0.5534	0.5804
Неважливі елементарні текстові елементи	0.7514	0.8251	0.7307	0.7285	0.7125	0.7386

Згідно з аналізом існуючих результатів, можемо стверджувати , що узгодженість відповідей експертів складає 0.7, що є достатнім для висновку, що результати експертної оцінки є узгодженими.

Основною задачею оцінки отриманого реферату є встановлення змістової відповідності між рефератом та первинним документом. Для вирішення даної задачі традиційно використовується критерій семантичної адекватності та семантичної еквівалентності [33, 34].Перший застосовується для оцінки точності реферування, а другий – для оцінки ступеня повноти відображення змісту первинного документу в рефераті. Для кількісної оцінки точності використовується відношення отриманих в рефераті релевантних елементарних текстових елементів до загальної кількості елементарних текстових елементів в рефераті. Для кількісної оцінки повноти використовується відношення отриманих в рефераті релевантних елементарних текстових елементів до загальної кількості релевантних елементарних текстових елементів. Існує наступна закономірність між цими параметрами: підвищення повноти веде до зменшення точності. Тому, окрім даних показників, використовується показник, що є гармонічним середнім параметрів повноти та точності, що обраховується за наступною формулою:

$$F = \frac{2 * \text{Точність} * \text{Повнота}}{\text{Точність} + \text{Повнота}}.$$

Наступним кроком проводився порівняльний аналіз результатів роботи розробленого методу з існуючими методами та алгоритмами на основі сукупності тестового набору текстів.

На сьогоднішній день відома лише одна система автоматизованого реферування україномовних текстів, що є вбудованою функцією офісного програмного забезпечення Microsoft Word під назвою MS Office AutoSummarize.

Необхідно наголосити, що існуюча система, обрана для порівняння, в якості фрагменту тексту використовує речення, а розроблена система використовує елементарні текстові елементи, то на етапі оцінки розробленого алгоритму результуючі елементарні текстові елементи були замінені реченнями, в яких вони містяться.

Результати порівняльної оцінки розробленого методу з традиційними методами представлені в табл. 4.5.

Таблиця 4.5

Середні значення показників якості методів автоматизованого реферування

Система (метод)	Повнота	Точність	F-параметр
Розроблена система (метод на основі аналізу функціональних відношень)	0.6481	0.6703	0.6603
СистемаMS Office AutoSummarize (метод на основі підрахунку статистичних показників)	0.3518	0.3275	0.3392

Згідно з аналізом отриманих результатів можна зробити висновок, що якість анотацій, отриманих за допомогою розробленого методу, є значно

вищим у порівнянні з рефератами, отриманими за допомогою традиційної статистичної системи.

Також необхідно визначити, як впливає обсяг тексту на якість реферату. Таке дослідження було проведено для текстів різного обсягу, але з однаковим коефіцієнтом зустрічаємості ключових фраз. Результати проведеного дослідженя представлені в табл. 4.6.

Таблиця 4.6

Залежність якості реферату від обсягу тексту

Текст	Кількість слів	Кількіть речень	Кількість ключових фраз	Коефіцієнт зустрічаємості ключових фраз	Повнота	Точність	F-параметр
1	130	8	5	0.62	0.6757	0.7353	0.7042
2	207	14	9	0.64	0.6216	0.7178	0.666
3	365	21	12	0.57	0.6923	0.6429	0.6666
4	515	32	18	0.56	0.6538	0.68	0.6666
5	750	45	26	0.57	0.6312	0.6541	0.6224

На основі аналізу отриманих результатів, можна сверджувати, що якість анотацій майже не залежить від обсягу тексту, якщо коефіцієнт зустрічаємості ключових фраз є майже однаковим. В наведених прикладах даний коефіцієнт є приблизно рівним 0.6.

Оскільки розроблений метод базується на аналізі ключових фраз , проведемо аналіз якості отриманих рефератів від коефіцієнту зустрічаємості ключових фраз. Результати проведеного аналізу представлені в табл. 4.7.

Таблиця 4.7

Залежність якості реферату від коефіцієнту зустрічаємості ключових фраз

Текст	Кількість слів	Кількіть речень	Кількість ключових фраз	Коефіцієнт зустрічаємості ключових фраз	Повнота	Точність	F-параметр
1	130	8	2	0.25	0.386	0.412	0.3984
2	150	10	3	0.3	0.431	0.446	0.4383
3	170	12	4	0.33	0.451	0.555	0.4978
4	230	16	7	0.44	0.581	0.565	0.5738
5	295	21	12	0.57	0.623	0.634	0.6286

На основі аналізу результатів можемо стверджувати, що якість реферату підвищується з підвищенням коефіцієнту зустрічаємості ключових фраз. Це пояснюється тим, що алгоритм базується на використанні функціональних відношень, що визначаються за допомогою ключових фраз.

4.5. Висновки до четвертого розділу

В рамках даного розділу розроблено програмне забезпечення для автоматизованого реферування україномовних науково-технічних текстів, що реалізоване з використання об’єктно-орієнтованого підходу, що дозволяє легко розширювати дану систему шляхом додавання нових модулів.

Також, проведений ряд експериментальних досліджень, що довели факт того, що якість рефератів, отриманих за допомогою даної системи, є в середньому на 20% вищою у порівнянні с рефератами, отриманими за допомогою традиційного методу, реалізованого у вбудованій функції пакету MS Office – AutoSummarize.

5. ПОБУДОВА БІЗНЕС-МОДЕЛІ

5.1. Опис проблеми

З розвитком інформаційних ресурсів Інтернет обсяг електронної науково-технічної інформації значно збільшився, внаслідок чого пошук необхідної інформації значно ускладнився. Особливо складним стало виділення необхідної інформації серед великих та самих по собі складних для прийняття документів, таких як наукові роботи. В даній ситуації актуальним шляхом вирішення даної проблеми є методи автоматизованого реферування тексту, що допомагають отримати стисле представлення текстових документів – рефератів. Звичайно, на сьогоднішній день існує безліч систем автоматичного реферування тексту, оскільки над вирішення даної проблеми почали шукати майже з самого початку розвитку ЄОМ, але більшість з подібних систем націлена на англійську мову. З визначених існуючих вирішень даної проблеми для української мови є лише вбудована в офісний пакет Microsoft Word функція Autosummarize. Однак, вона працює на базі статистичних методів, тобто результатом роботи даної функції буде лінійний набір слів, згідно їх частоти появи або ваги, саме тому сенс отриманого реферату може не в повній мірі або взагалі некоректно відображати початковий текст.

Звичайно, з даною задачею набагато краще впорається людина самостійно, але для створення реферату малого обсягу та із збереженням основної мети тексту, повинні працювати експерти в даній області або автор тексту, оскільки мова йдеться про науково-технічні тексти, не кажучи вже про те, що робота експертів даного рівня є дорогою та незручною для користувача, оскільки, скоріш за все, доведеться стояти в черзі на виконання своєї роботи.

Саме через цю низку вразливостей виникла необхідність автоматизованого реферування україномовних науково-технічних текстів. Усі недоліки описані вище узагальнені на рис. 5.1 «Дерево проблем».

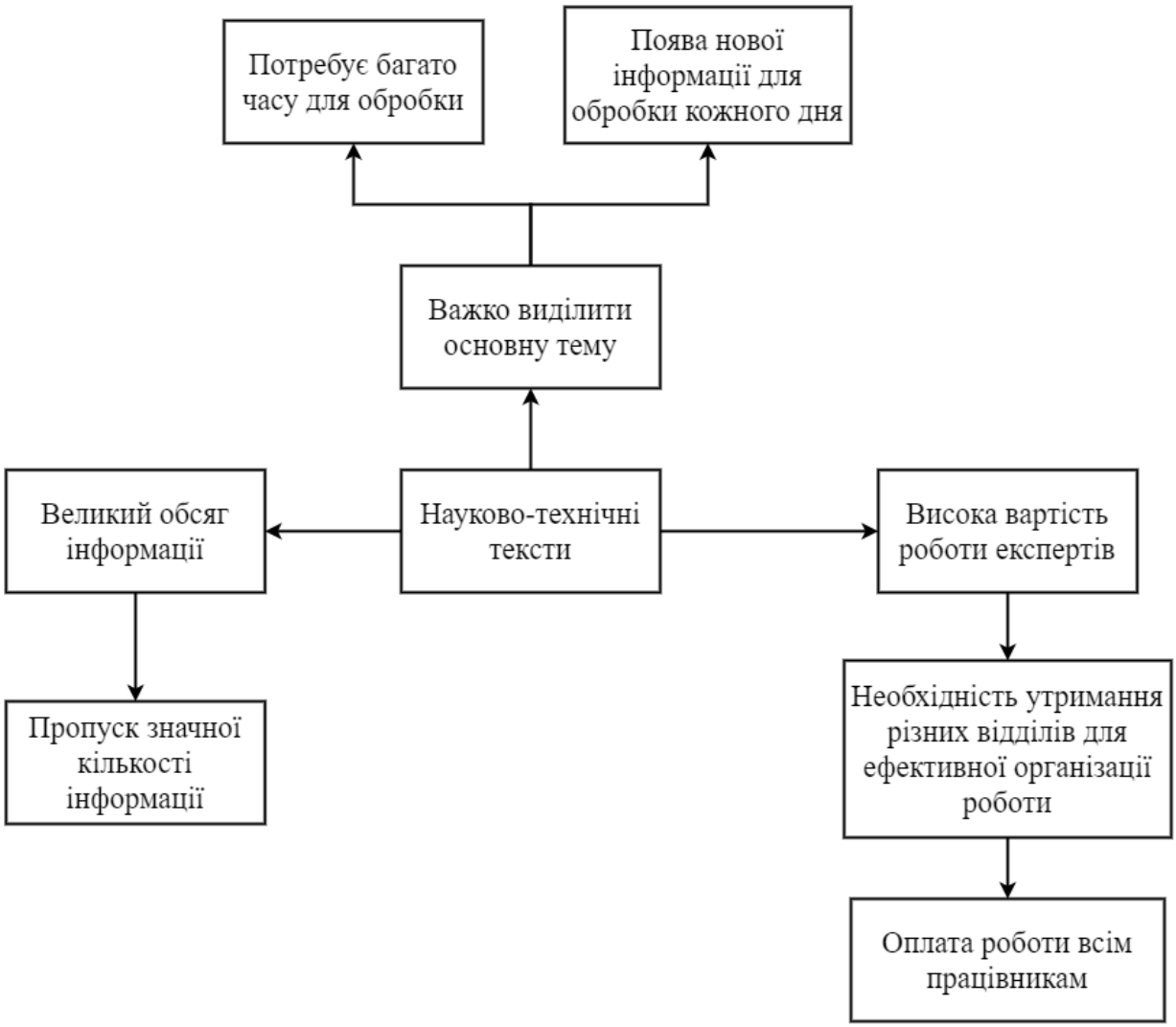


Рис. 5.1. Дерево проблем

5.2. Зацікавлені сторони

Зацікавленими сторонами у описаних в минулому підрозділі проблем є багато зацікавлених сторін. В першу чергу, це звичайно, користувачі мережі Інтернет, шо шукають певні науково-технічні роботи при аналізі потрібної їм предметної галузі при розробці власних досліджень.

До того ж, багато наукових конференцій заздалегідь викладають розклад майбутніх виступів, але навряд тема виступу з прізвищами лекторам може надати початківцям в науковій галузі, чи буде дана конференція корисною для них, тому втрачають більшу частину конференцій. З іншого боку, якщо організатори даних наукових

конференцій будуть мати змогу викласти дуже стислий опис інформації, наведеної в статтях або тезах на своїх веб-сайтах – це б допомогло організатором конференцій отримувати більшу кількість учасників.

Усі зацікавлені сторони, їх вплив та інтереси наведені у табл.5.1.

Таблиця 5.1

Зацікавлені сторони

Зацікавлена сторона	Інтерес зацікавленої сторони	Вплив зацікавленої сторони	Стратегія приваблення зацікавлених сторін
Організатори наукових конференцій	Можливість залучити більшу кількість учасників	Високий	Проведення презентації для представників зацікавлених осіб. Участь у спеціалізованих конференціях і форумах
Наукові дослідники, що шукають інформацію в мережі Інтернет	Можливість якісніше обирати підходящу інформацію	Середній	

5.3. Комерційне рішення. Основні характеристики

Базуючись на описаних раніше проблемах, кінцева система повинна реалізовувати автоматизований метод реферування україномовних текстів, враховуючи особливості природномовних текстів. Оскільки мі орієнтуємось на організаторів конференцій, які повинні мати змогу отримувати дуже стислий реферат з вхідного тексту, а для наукових дослідників параметр стислості може бути різним, то вказане програмне забезпечення повинее приймати в якості параметру бажаний обсяг отриманого реферату.

Також слід зазначити, що для збереження інформативності

отриманого реферату, система повинна використовувати метод, що враховує особливості структури тексту.

5.4. Конкурентні переваги рішення

Як наголошували раніше, вирішення даної проблеми досліджувалось вже давно, перші роботи, присвячені вирішенню даної задачі були опубліковані понад 60 років тому, але більшість з них розглядала лише англійську мову. Згодом, з'явились рішення і для інших мов, але на жаль, для української мови існує лише одна система, яка базується на статистичних методах. До того ж, розробка систем загального призначення, що враховують нелінійну структуру природномовних даних, майже неможливо розробити, оскільки кожен стиль тексту має свої характерні особливості. Звичайно, найкращий реферат можна отримати лише за допомогою експертів в потрібній галузі, але таких підхід має дуже багато недоліків, які описані в минулих підрозділах. Розроблена система має задовольнити потреби зацікавлений сторін, що викладені в минулому підрозділі. Програмний продукт, якій був запропонований, має такі конкурентні переваги:

- зменшення витрат за рахунок відмови від залучення висококваліфікованих співробітників;
- покращення точності реферату за рахунок використання методу, що враховує особливості структури тексту;
- можливість задати бажаний обсяг реферату.

5.5. Клієнти. Сегменти ринку споживання

Головні клієнти даної системи автоматизованого реферування науково-технічних текстів – це, в першу чергу організатори великих конференцій, які досить часто пропонують науковцям публікацію доповідей при умові виступу та запрошують охочих бути присутніми на лекціях доповідачів. З метою зацікавити початкових науковців, які не можуть бути

впевненими, чи відповідають теми доповідей їх сфері обізнаності. Навіть якщо використовувати існуючі системи автоматизованого реферування текстів для англійської мови, оскільки у більшості випадків додатково надсилаються тези англійською мовою, та використовувати лише даний тип опису текстів, все одно існує декілька недоліків. По-перше, як було вказано, системи загального призначення навряд зможуть якісно визначити важливі елементи специфічних та насичених термінами науково-технічних текстів. По друге, для початкових науковці, що не дуже добре знають англійську мову, така інформація буде марною.

5.6. Унікальна ціннісна пропозиція

Ціннісна пропозиція – це усі переваги, які отримує споживач, вибираючи послуги або товар. Низка проблем була виділена у дереві проблем і інтереси зацікавлених сторін були проаналізовані. Організатори конференцій хочуть залучити більше відвідувачів, а дослідники, що аналізують предметну галузь, хочуть простіше відсіювати нерелевантну інформацію. Навіть якщо програмне забезпечення не зможе забезпечити гарний показник швидкодії, дане програмне забезпечення дозволить значно полегшити процес розуміння основної мети наукової публікації.

Спираючись на все вищесказане можна виокремити унікальну ціннісну пропозицію, якою є розроблений метод автоматизованого реферування україномовних науково-технічних текстів, який враховує нелінійну та ієрархічну структуру тексту, що є основною перевагою.

5.7. Доходи та витрати

Користувач буде мати доступ до програмного продукту після придбання ліцензії на продукт.

Виділяють такі типи ліцензій програмного забезпечення:

1. Freeware. Вільно та безкоштовно розповсюджувані повнофункціональні програми. Купувати подібні програми не потрібно. Вони, як правило, поширюються через середу Інтернет

або в якості доповнення з платними комерційними продуктами.

2. Shareware. Умовно-безкоштовне програмне забезпечення – користувачеві безкоштовно надається програмне забезпечення неповного функціоналу, тобто з деякими обмеженнями, що діють до тих пір, поки не буде проведена оплата за повнофункціональний продукт. Обмеження можуть бути:
   – функціональними, тобто користувачеві доступні не всі функціональні можливості – це так звані демо-версії (demo);
   – тимчасовими, тобто без оплати продукт в повному функціоналі працює якийсь календарний час або певну кількість запусків – пробні версії (trial).
3. Public domain software. Даний тип ліцензій схожий на freeware – програмні продукти цього типу також поширюються безкоштовно. Однак, на відміну від freeware, де за автором програми зберігаються всі права на неї, у випадку з public domain у автора ці права відсутні. Програмне забезпечення поширюється разом з вихідним кодом, і автор, викладаючи свій продукт в загальний доступ, повністю відмовляється від своїх прав. Поширення такого роду ліцензій було характерно для початку масового доступу в Інтернет. У даний час продукти з цим типом ліцензії практично не випускаються
4. Open Source. Програмне забезпечення поширюється на безкоштовній основі разом з вихідним кодом. Однак автор уже не відмовляється від своїх прав. Існує міжнародна система вимог до ліцензії на програмний продукт, який називається The Open Source Definition (OSD). Модифіковане будь-яким співавтором забезпечення повинно поширюватися на тих же умовах, що і вихідний продукт – тобто модифіковане ПО не можна перевести в платне і комерційне.
5. Commercial. Комерційне програмне забезпечення, тобто

програмне забезпечення, завжди розповсюджується тільки за плату. Оплата повинна бути здійснена авансом або відразу після отримання копії на ліцензійному диску або дискеті у фірмовій упаковці.

5.8. Бізнес-модель

Складемо тепер бізнес-модель, узагальнивши усе вище написане і перенесемо на канву, яка зображена на рис. 5.2.

Споживачі:

- організатори наукових конференцій;
- наукові дослідники, що шукають інформацію в мережі Інтернет.

Проблема:

- існуючі системи автоматизованого реферування україномовних науково-технічних текстів не задовольняють проблемам інформативності отриманого реферату;
- людські послуги занадто дорогі для вирішення такого роду завдання;
- обсяги інформації занадто великі.

Рішення: система автоматизованого реферування україномовних науково-технічних текстів, що враховує особливості природномовних текстів.

Унікальна ціннісна пропозиція:

- програмний застосунок, що реалізує метод автоматизованого реферування тексту з врахуванням нелінійної та ієрархічної структури текстових даних;
- допомога організаторам конференцій у залученні додаткових відвідувачів.

Потоки доходів:

- продаж ліцензій;

Структура витрат:

– заробітна плата працівникам;
– оренда невеликого приміщення;
– покупка обладнання;
– послуги юриста;
– послуги бухгалтера;
– податки.

Взаємовідносини з клієнтами:

– проведення презентацій, участь у спеціалізованих конференціях і форумах;
– технічна підтримка.

Канали: реклама в мережі Інтернет, відділи співпраці на конференціях.

Ключові метрики: кількість ліцензій, які були продані.

Таблиця 5.2

Канва бізнес-моделі

<table>
<tr><td rowspan="2">Проблема
1. Наявні рішення не задовільнять потребам.
2. Людські послуги занадто дорогі.
3. Обсяги інформації занадто великі.</td><td>Рішення
Система автоматизованого реферування україномовних науково-технічних текстів, що враховує особливості природномовних текстів</td><td rowspan="2">Унікальна ціннісна пропозиція
1.Програмний застосунок, що реалізує метод автоматизованого реферування тексту з врахуванням нелінійної та ієрархічної структури текстових даних;
2.Допомога організаторам конференцій у залученні додаткових відвідувачів</td><td>Взаємодія з клієнтами
1. Проведення презентацій, участь у спеціалізованих конференціях і форумах
2. Технічна підтримка</td><td rowspan="2">Споживачі
1.Організатори наукових конференцій;
2.Наукові дослідники, що шукають інформацію в мережі Інтернет</td></tr>
<tr><td>Ключові метрики
Кількість ліцензій, що були продані</td><td>Канали
Реклама в мережі Інтернет, відділи співпраці на конференціях</td></tr>
<tr><td colspan="3">Структура витрат
1. Заробітна плата. 2. Оренда приміщення
3. Податкові витрати. 4. Послуги юриста</td><td colspan="2">Потоки доходів
Доходи від продажу ліцензій</td></tr>
</table>

5.9. Висновки до п’ятого розділу

У даному розділі проаналізовано поточну ситуацію у сфері автоматизованого реферування україномовних науково-технічних текстів.

Створено дерево проблем з усіма наявними перешкодами у цій сфері. Також були описані всі зацікавлені сторони, їх вплив та зацікавленість. В результаті цього була сформульована унікальна ціннісна пропозиція програмного продукту. Були проаналізовані сегменти ринку споживання та основні клієнти.

Крім цього були визначені потоки доходів та витрат.

Була побудована бізнес-модель, яка описує, як продукт створює цінність і може заробити на цьому.

ВИСНОВКИ

У даній магістерській дисертаційній роботі виконаний аналіз існуючих підходів автоматизованого реферування тексту. Визначено, що для україномовних текстів до практичної реалізації дійшли лише статистичні методи, які не враховують нелінійну та ієрархічну структуру тексту.

Запропонований метод формалізованого опису структури україномовних науково-технічних текстів, що відрізняється врахуванням нелінійної та ієрархічної структури тексту, що дозволяє підвищити якість автоматизованого реферування україномовних науково-технічних текстів. Розроблений критерій коректності структури тексту, виконані формалізація характеристик та обмежень для коректних текстових структур, що є розширенням основних положень теорії риторичної структури.

Розроблений метод визначення функціональних відношень між фрагментами тексту на основі аналізу ключових фраз української мови, який відрізняється використанням вузьконаправленого словника ключових фраз, що дозволяє зменшити інформаційну надлишковість за рахунок відмови від словників загального призначення.

Розроблений алгоритм побудови структури тексту на основі множини визначених функціональних відношень між фрагментами тексту, який відрізняється врахуванням неоднозначності відношень з ключових фраз, генерацією альтернативних множин варіантів коректних структур тексту та вибір оптимальної за критерієм сукупної метрики.

Проведена експериментальна перевірка запропонованих методів та алгоритмів, що реалізовані в розробленому програмного забезпечення автоматизованого реферування україномовних науково-технічних текстів. Згідно з проведеними дослідженнями, можна стверджувати що якість є в середньому на 20% вищою у порівнянні с рефератами, отриманими за допомогою традиційного методу, реалізованого у вбудованій функції пакету MS Office – AutoSummarize.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. The automatic creation of literature abstracts. [Текст] / H.P.Luhn – IBM Journal of Research and Development, 1958. – С. 159-165.
2. Севбо, И.П. Структура связного текста и автоматизация реферирования [Текст] / И.П. Севбо – М.: Наука, 1969. – 135 с.
3. Берзон, В.Е. Синтаксические сверхфразовые связи и их инженерно-лингвистическое моделирование [Текст]: / В.Е. Берзон, Э. М. Добрускина. – Кишинев: Штиинца, 1986. – 168 с.
4. Скороходько, Э.Ф. Семантические сети и автоматическая обработка текста [Текст]: / Э.Ф. Скороходько. – Київ: Наукова думка,1983. – 219 с.
5. Афонин А.А., Леонов, В.П. Реферирование и аннотирование научно-технической литературы [Текст] / В.П. Леонов. – Новосибирск: Наука, 1986. – 175 с.
6. Рождественская, Н.В. Дискурс как высшая единица коммуникативного акта. / Н.В. Рождественская – Вісник Запорізького державного університету, 2002. – с. 10–15.
7. Леонов, В.П. Реферирование и аннотирование научно-технической литературы [Текст] / В.П. Леонов. – Новосибирск:Наука, 1986. – 175 с.
8. Кулагина, О.С. Исследования по машинному переводу [Текст]: / О. С. Кулагина. – М.: Наука, 1979. – 320 с.
9. Luhn, H.P. The automatic creation of literature abstracts. [Текст] / H.P. Luhn – IBM Journal of Research and Development, 1958. – С. 159-165.
10. Пиотровский, Р.Г. Текст, машина, человек [Текст]: / Р.Г. Пиотровский. – Л.: Наука, 1975. - 327с.
11. Визуализация семантической структуры и реферирование текстов на естественном языке [Електронний ресурс] – Режим доступу : https://docplayer.ru/42896123-Vizualizaciya-semanticheskoy-struktury-i-referirovanie-tekstov-na-estestvennom-yazyke.html

12. Автоматическое реферирование веб-документов с учетом запроса [Електронний ресурс] – Режим доступу : https://nsu.ru/xmlui/handle/nsu/8971
13. Яцко, В.А. Симметричное реферирование: теоретические основы и методика [Текст] / В.А. Яцко – НТИ, 2002. – С. 18-28.
14. Intelligent Miner for Text [Електронний ресурс] – Режим доступу : https://www-01.ibm.com/common/ssi/cgi-bin/ssialias?infotype=an&subtype=ca&htmlfid=897/ENUS298-061&language=enus&appname=skmwww
15. MS Office AutoSummarize [Електронний ресурс] – Режим доступу : https://www.groovypost.com/howto/summarize-articles-microsoft-word/
16. TextAnalyst [Електронний ресурс] – Режим доступу : http://www.analyst.ru/index.php?lang=rus&dir=content/products/
17. Oracle Text [Електронний ресурс] – Режим доступу : https://docs.oracle.com/cd/E24693_01/text.11203/e24435/query.htm
18. Copernic summarizer [Електронний ресурс] – Режим доступу : https://www.copernic.com/data/pdf/copernic-summarizer-specification-sheet.pdf
19. Макаров, М.Л. Основы теории дискурса [Текст] : / М.Л.Макаров. – М.: Гнозис, 2003. – 280 с.
20. Barzilay, M. Using lexical chains for text summarization. [Текст] / M.Barzilay, R. Elhadad – In Proceedings of the ACL/EACL-97 Workshop on Intelligent Scalable Text Summarization. – Madrid, Spain, 1997. – С. 10-17.
21. Grimes, J.E. The Thread of Discourse. [Текст] / J.E. Grimes – The Hauge, Paris: Moution. – 1975. – 346 с.
22. Grosz, B.J. Attention, intentions, and structure of discourse [Текст] / B.J. Grosz, C.L. Sidner – Computational Linguistics, 1986 – С. 203-226.
23. Grimes, J.E. The Thread of Discourse. [Текст] / J.E. Grimes – The Hauge, Paris: Moution. – 1975. – 346 с.

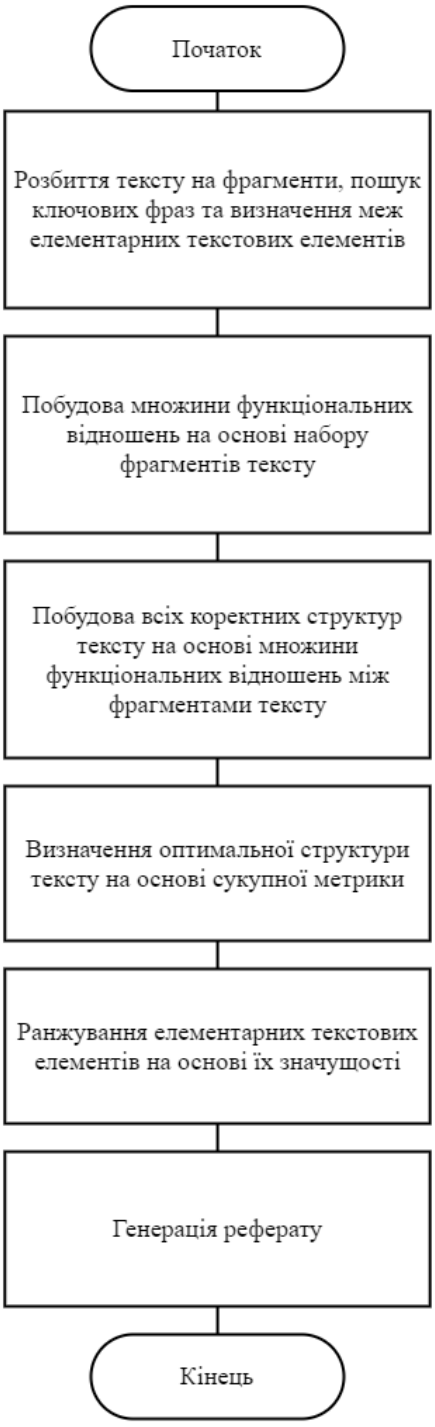
24. Haliday, A.K. Cohesion in English. [Текст] / A.K. Haliday, R. Hasan – England, Londin: Longman, 1976. – 221 с.
25. Лукашевич, Н.В Представление знаний в системе автоматической обработки текстов [Текст] / Н.В. Лукашевич, А.Д. Салий – НТИ, 1997. – С. 15-23.
26. Марчук, Ю.Н. Компьютерная лингвистика [Текст] / Ю.Н. Марчук – АСТ, Восток-Запад, 2007. - 226 с.
27. Мельчук, И.А Опыт теории лингвистических моделей «Смысл-Текст». Семантика, синтаксис [Текст] / И.А.Мельчук. – М.:Наука, 1999. – 314 с.
28. Горский, В.Г., Метод согласования кластеризованных ранжировок [Текст] / В. Г. Горский, А.И. Орлов, А.А. Гриценко – Автоматика и телемеханика, 2000. – С. 59-167.
29. Системний аналіз та системне проектування складних систем [Електронний ресурс] – Режим доступу : https://studfile.net/preview/9375743/page:4/
30. Об'єктно - орієнтоване програмування [Електронний ресурс] – Режим доступу : http://programming.in.ua/programming/basisprogramming/25-oop.html
31. Java [Електронний ресурс] – Режим доступу : https://uk.wikipedia.org/wiki/Java
32. Коефіцієнт узгодженості Кенделла Електронний ресурс] – Режим доступу : http://statistica.ru/local-portals/quality-control/element-5/
33. Inderjeet, M. Advances in automatic text summarization [Текст] / M. Inderjeet, M. T. Maybury – The MIT Press, 1999. – 434 с.
34. Rijsbergen, C. J. Information retrieval. [Текст] / C. J. Rijsbergen. – Butterworths, London, 1979. – 325 с.

ДОДАТКИ

Додаток 1

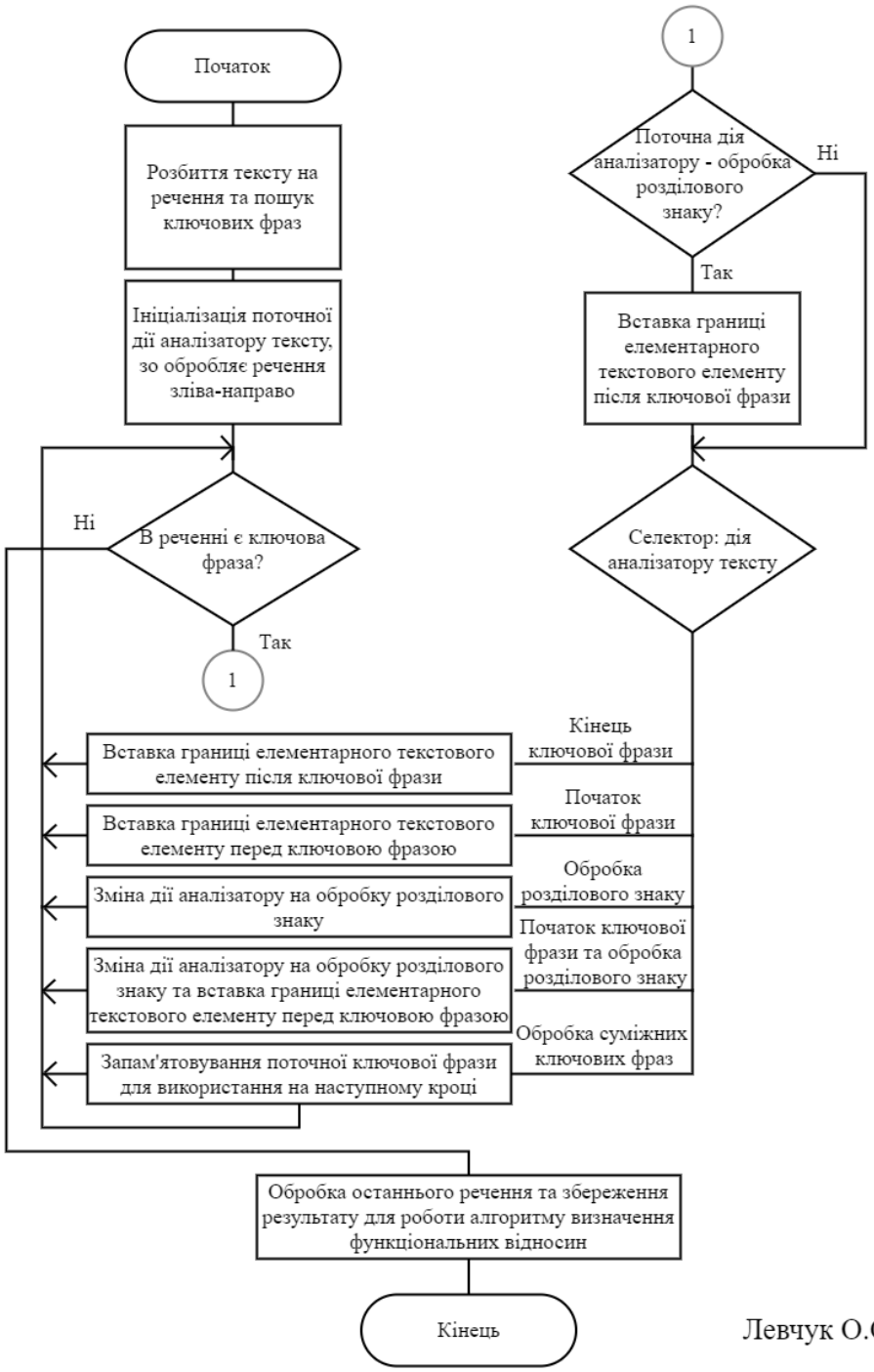
Копії графічних матеріалів

УЗАГАЛЬНЕНИЙ АЛГОРИТМ АВТОМАТИЗОВАНОГО РЕФЕРУВАННЯ



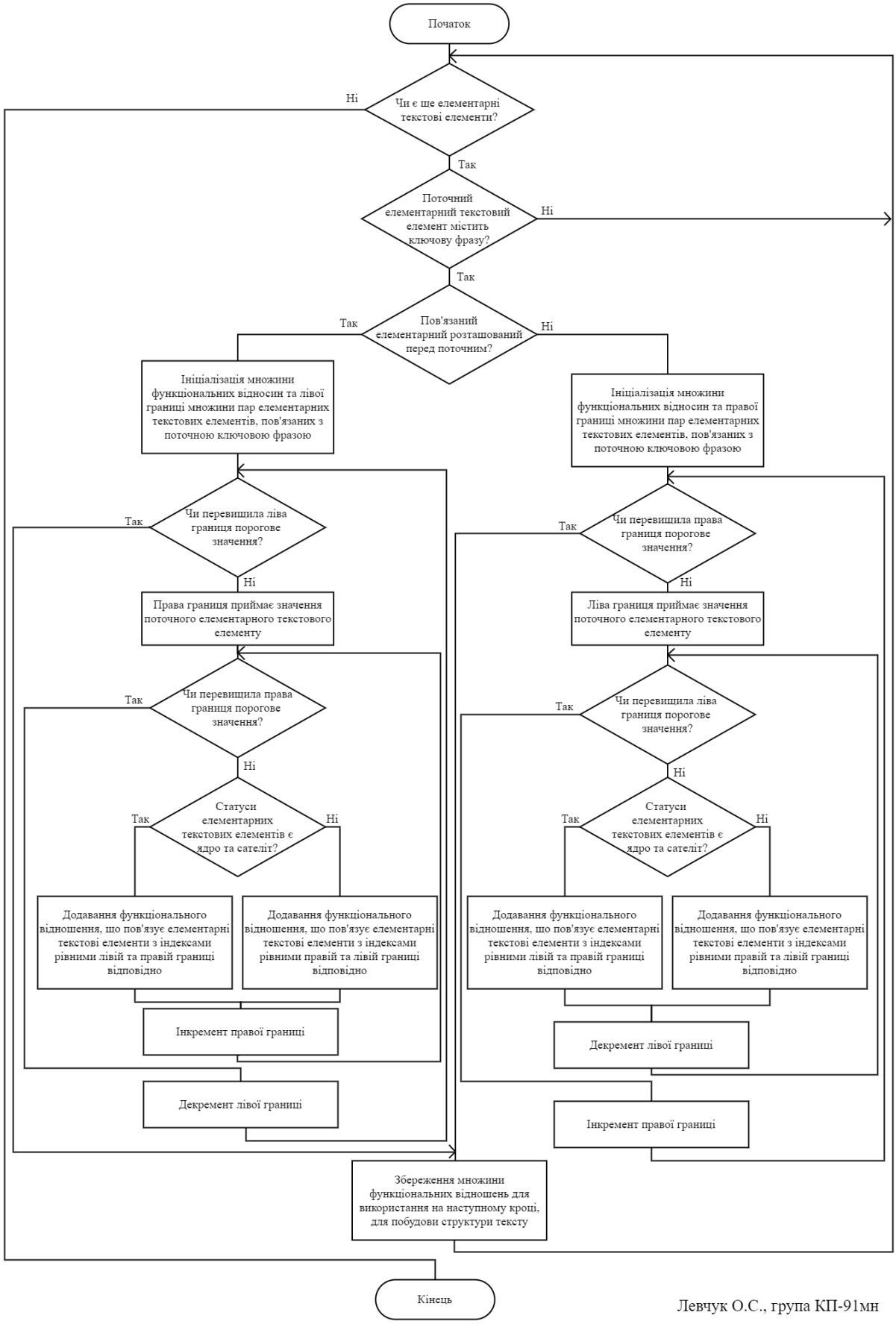
Левчук О.С., група КП-91мн

АЛГОРИТМ ВИЗНАЧЕННЯ ГРАНИЦЬ ЕЛЕМЕНТАРНИХ ТЕКСТОВИХ ЕЛЕМЕНТІВ



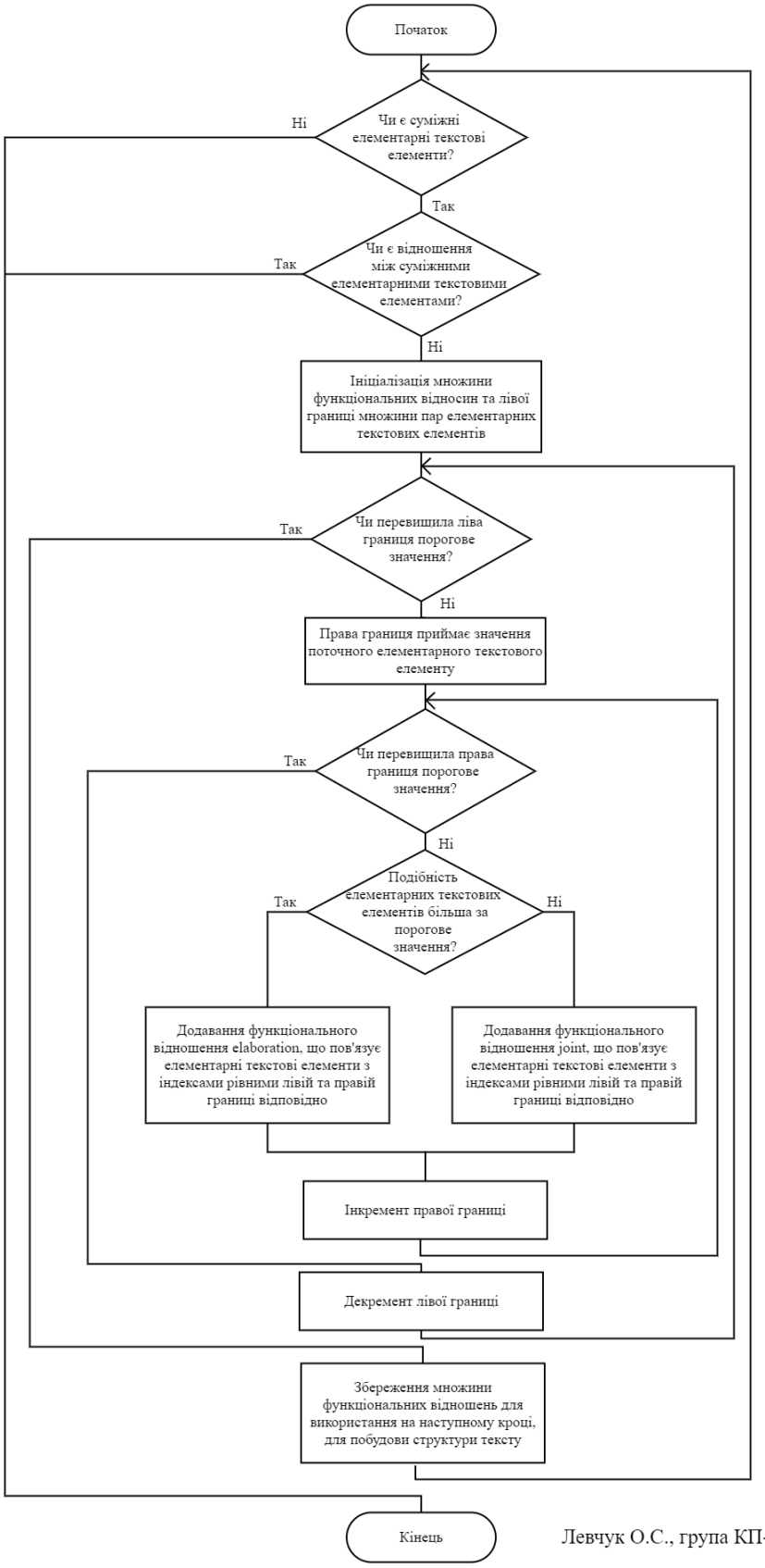
Левчук О.С., група КП-91мн

АЛГОРИТМ ПОБУДОВИ МНОЖИНИ ФУНКЦІОНАЛЬНИХ ВІДНОШЕНЬ НА ОСНОВІ НАБОРУ ЕЛЕМЕНТАРНИХ ТЕКСТОВИХ ЕЛЕМЕНТІВ



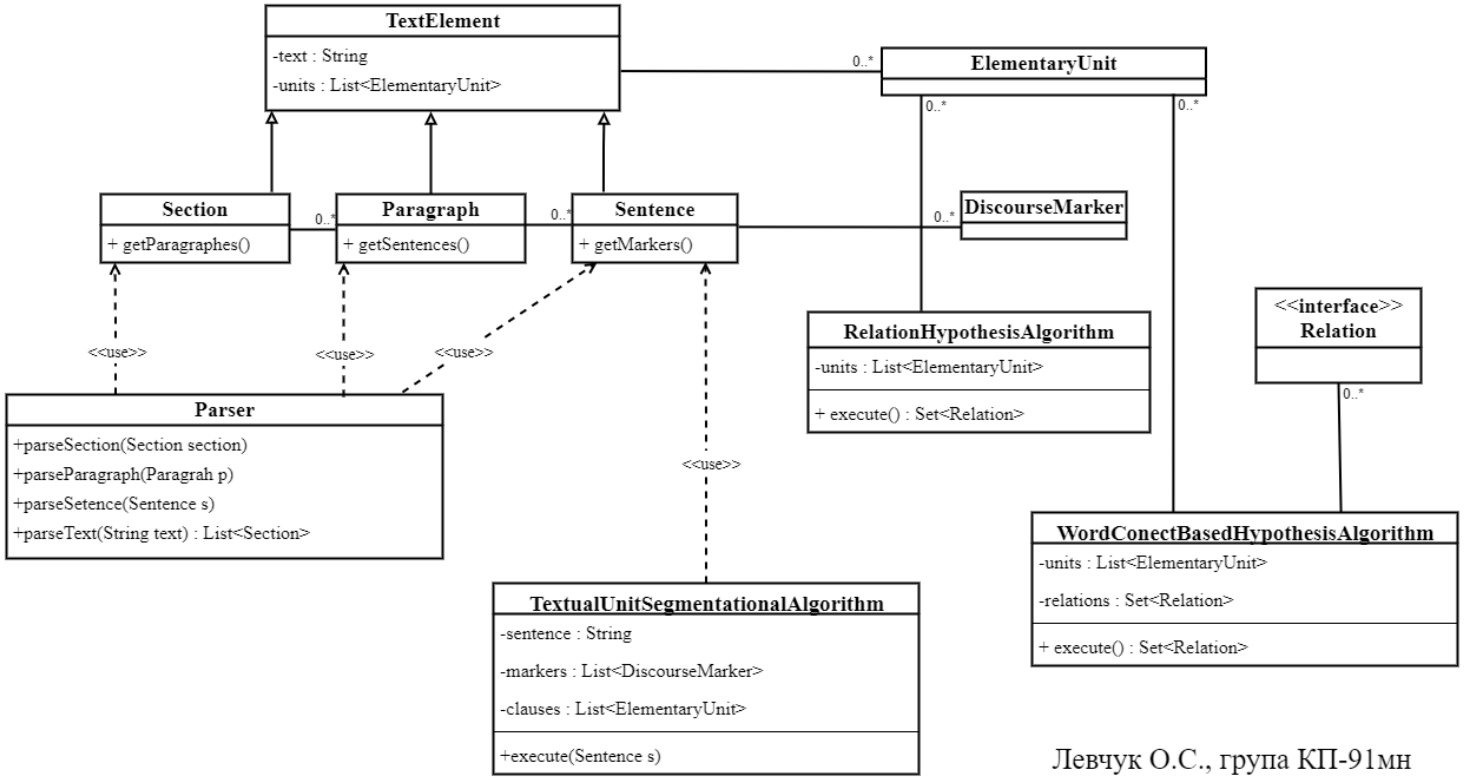
Левчук О.С., група КП-91мн

АЛГОРИТМ ВИЗНАЧЕННЯ ФУНКЦІОНАЛЬНИХ ВІДНОШЕНЬ ДЛЯ НЕПОВ'ЯЗАНИХ ЕЛЕМЕНТАРНИХ ТЕКСТОВИХ ЕЛЕМЕНТІВ



Левчук О.С., група КП-91мн

ДІАГРАМА КЛАСІВ ПІДСИСТЕМИ АНАЛІЗУ ТЕКСТУ



Левчук О.С., група КП-91мн

СТРУКТУРА СИСТЕМИ АВТОМАТИЗОВАНОГО РЕФЕРУВАННЯ

Левчук О.С., група КП-91мн

Додаток 2

Лістинг

```
public abstract class BaseTreeAxiom implements Axiom {

      protected TreeAxiom axioml;
      protected TreeAxiom axiom2;
      protected Relation relation;
      public BaseTreeAxiom(TreeAxiom al, TreeAxiom a2, Relation relation)
      {      this.axioml = al;
            this.axiom2 = a2;
            this.relation = relation;
      }
      abstract public boolean apply();
      abstract public Axiom derive();
      public TreeAxiom getAxiomlO { return axioml;}
      public TreeAxiom getAxiom2() {
      return axiom2;
}

public abstract class BaseSimpleAxiom extends BaseTreeAxiom

{

      public BaseSimpleAxiom(TreeAxiom al, TreeAxiom a2, Relation relation)
       { super(al, a2, relation);

      protected boolean apply(RhetoricalRelationStatus statusl,
      RhetoricalRelationStatus sta-tus2, boolean isHypotactic) {

            return axioml.getTreeStatusO.equals(status) &&
      axiom2.getTreeStatus().equals(status2)&&
      SimpleAxioms.rhet_rel(axioml, axiom2, (SimpleRhetorical Relation)
      relation) && (isHypotactic ? SimpleAxioms.hypotactic(relation):
      SimpleAxioms.paratactic(relation));

      protected Axiom derive(Rhetorical Relationstatus status, Set<lnteger>
promotion) { BinaryTree tree = new BinaryTree();

      TreeNode node = new TreeNode(0, relation.getRelation(), status,
promotion);

      node.setLeft(axioml.getTree() .getRootO);

      node.setRight(axiom2.getTree().getRootO);

      tree.setRoot(node);

      return new TreeAxiom(tree, axioml.getLeftO, axiom2.getRight(),
Setutils.extract(Setuti1s.i ntersect(axioml.qetRelations(), axiom2
.getRelationsO), relation));

}

public class Simple_N_N_N extends BaseSimpleAxiom {

      public Simple_N_N_N(TreeAxiom al, TreeAxiom a2, Relation relation) {
      super(al, a2, relation);

      @Override

      public boolean apply() {
```

```
        return apply (RhetoricalRelationstatus.NUCLEUS, Rhetorical
        Relationstatus.NUCLEUS, false);

        @Override

        .public Axiom derive() {

        return derive(RhetoricalRelationstatus.nucleus, setIsunion(axioml.
        getPromoti on () , axiom2.getPrornotion()));

        public class Simple_N_N_s extends BaseSimpleAxiom {

        public Simple_N_N_s(TreeAxiom al, TreeAxiom a2, Relation relation) {

        super(al, a2, relation);

        @Override

        public boolean apply() {

        return apply(RhetoricalRelationstatus.NUCLEUS,

        calRelationstatus.NUCLEUS, false);

        @Override

        public Axiom derive() {

        return derive(RhetoricalRelationstatus.SATELLITE,
        SetIsuunion(axioml.getPromotion () , axiom2.getPromotion()));

}

public class simple_N_s_N extends BaseSimpleAxiom {

        public Simple_N_S_N(TreeAxiom al, TreeAxiom a2, Relation relation) {
        super(al, a2, relation);

        @override

        public boolean applyO {

        return apply(Rhetori calRelationstatus.NUCLEUS,

        RhetoricalRelationstatus.SATELLITE, true);

        @Override

        public Axiom derive() {

        return derive(RhetoricalRelationstatus.nucleus, axioml.
        getPromotionO);

        public class Simple_N_S_S extends BaseSimpleAxiom {

        public Simple_N_S_S(TreeAxiom al, TreeAxiom a2, Relation relation) {
        super(al, a2, relation);

        @Override

        public Axiom derived {
```

```
        return derive(RhetoricalRelationstatus.SATELLITE, axioml.getPromotion
        () );

}

public class Simple_S_N_N extends BaseSimpleAxiom {

        public Simple_S_N_N(TreeAxiom al, TreeAxiom a2, Relation relation) {
        super(al, a2, relation);

        @override

        public boolean apply () {

        return apply(RhetoricalRelationstatus.SATELLITE,

        RhetoricalRelationStatus.NUCLEUS, true);

        @Override

        public Axiom derived {

        return derive(RhetoricalRelationstatus.NUCLEUS,
        axiom2.getPromotion());

        public class simple_s_N_S extends BaseSimpleAxiom {

        public Simple_s_N_s(TreeAxiom al, TreeAxiom a2, Relation relation) {
        super(al, a2, relation);

        @Override public boolean applyd { return
        apply(RhetoricalRelationstatus.satellite,
        RhetoricalRelationstatus.NUCLEUS, true) ; }

        @Override public Axiom derived { return
        derive(RhetoricalRelationstatus.SATELLITE, axiom2.getPromotion());

        }

}

public class ProofTheoryAlgorithm implements ProofTheoryApproach {

        private List<ElementarryDiscourseUnit> units;

        private Set<Relation> relations;

        public proofTheoryAlgorithm(Li st<El ementarryDi scou rseuni t> uni
        ts, Set<Relation> relations) {

        this.units = units;

        this.relations = relations;

        public Set<Axiom> executed throws NoRelationsException{ if (isimpl
        eAxioms .hoid(relations)) {

        throw new NoRelationsException("There were no relations defined");

        }
```

```
Set<Axiom> axioms = new HashSet<Axiom>();

for (ElementarryDiscourseUnit unit : units) {
generateTreeLeafAxiom(axioms, unit, relations);

final int N = units.sized;

final int delta = units.get(O) .getlndexd - 1;

for (int size_of_span = 1; size_of_span <= (N - 1); size_of_span++) {

for (int 1 =1; 1 <= (N - size_of_span); 1++) {

int h = 1 + size_of_span;

for (int b = 1; b <= (h - 1); b++) {

Set<TreeAxiom> treeAxiomsl = findTreeAxioms(l + delta, b + delta,
axioms);

Set<TreeAxiom> treeAxioms2 = findTreeAxioms(b + 1 + delta, h + delta,
axioms);

for (TreeAxiom axioml : treeAxiomsl) {

for (TreeAxiom axiom2 : treeAxioms2) {

Set<? extends Relation> relations =

findTreeAxiomRelations(axioml, axiom2);

for (Relation relation : relations) {

if(axioml.getRight() == (axiom2.getLeft() - 1)) {

if (relation instanceof SimpleRhetorical Relation) {

applySimpleAxioms(axioml, axiom2, relation, axioms); Jelse {

applyExtendedAxioms(axioml, axiom2, relation, axioms);

}}}}}} return axioms;

}

private void generateTreeLeafAxiom(Set<Axiom> axioms,
ElementarryDiscourseUnit unit, Set<Relation> relations) {

int index = unit.getlndexd;

BinaryTree tree = new BinaryTree();

tree.i nsert(i ndex, RhetoricalRelati on.leaf, Rhetori calRelati
onstatus.NUCLEUS, unit, index);

axioms.add(new TreeAxiom(tree, index, index, relations));

tree = new BinaryTreed;

tree.insert(index, RhetoricalRelation.leaf,
RhetoricalRelationstatus.SATELLITE, unit, index);
```

```
      axioms.add(new TreeAxiom(tree, index, index, relations));

      }

      private void applyTreeAxiom(Axiom axiom, Set<Axiom> axioms) { if
      (axiom.applyO) {

      axioms.add(axiom.derived);

}

public class Parser {

      public void parseSection(Section section) {

            String paragraphpattern = corpusDAO.getParagraphPattern();

            Pattern regexp = null; try {

            regexp = Pattern.compile(paragraphPattern);

            catch(PatternSyntaxException e) {

            System.err.println("Pattern is invalid: " + paragraphpattern);
            return;

            Matcher matcher = regexp.matcher(section.getTextO);

            int prevoffset = 0;

            while(matcher.find()) {

            int offset = matcher.start();

            if (offset > 0) {

            String text = section.getTextO.substring(prevoffset, offset +
            matcher.group().length()) ;

            }

            ection.addParagraph(new Paragraph(text));

            prevOffset = offset + matcher.groupO .lengthO ;

            String text = section.getTextQ. substring(prevoffset);

            section.addParagraph(new Paragraph(text));

            1

            public void parseParagraph(Paragraph paragraph) {

            System.out.println("Parsing paragraph: " + paragraph.getTextQ);

            String sentencepattern = CorpusDAO.getSentencePatternQ;

            Pattern regexp = null; try {

            regexp = Pattern.compile(sentencepattern);

            catch(PatternSyntaxException e) {
```

```
            System.err.printlnC'Pattern is invalid: " + sentencepattern);
            return;

            }

      Matcher matcher = regexp.matcher(paragraph.getTextQ);

      int prevoffset = 0;

      while(matcher.findQ) { int offset = matcher.startQ; if (offset > 0) {

      String text = paragraph.getTextQ.substringfprevoffset, offset +
matcher.groupQ .lengthQ) ;

      paragraph.addSentence(new Sentence(text)); prevoffset = offset +
matcher .groupQ . 1 engthQ ;

      } 1

      String text = paragraph.getTextQ.substring(prevoffset).trimQ;

      if (! ".equals(text)) paragraph.addsentence(new Sentence(text));

      public void parseSentence(Sentence sentence) {

      System.out.println("parsing sentence: '' + sentence.getTextQ) ;

      String patterns = CorpusDAO. getAl 1 PatternsQ ;

      List<DiscourseMarker> markers = new ArrayList<DiscourseMarker>();

      for (String pattern : patterns) { Pattern regexp = null; try {

      regexp = Pattern.compile(pattern);

      catch(PatternSyntaxException e) {

      System.err.printlnC'Pattern is invalid: " + pattern); continue;

      Matcher matcher = regexp.matcher(sentence.getTextQ);

      while(matcher.find()) {

      int offset = matcher.startQ;  .

      markers.addAll(CorpusDAO.getDiscourseMarkers(pattern, offset,
matcher.groupQ)) ;

      Col1ections.sort(markers, DiscourseMarker.MARKER-COMPARATOR);
sentence.setMarkers(markers.toArray(new DiscourseMarker {})); for
(DiscourseMarker marker : markers) {

      System.out.println("Found: " + marker.getTextQ + " start: " + mark-
er.getOffsetQ);

      } }

      public List<Section> parseText(String text) {

            Section section = new Section(text);
```

```
            parseSection(section);

            for (Paragraph paragraph : section.getParagraphes()) {
      parseParagraph(paragraph);

            for (Sentence sentence : paragraph.getSentencesQ) {
      parsesentence(sentence);

            return Arrays.asList(section);

            }

}

public class Main {

      public static void main(String args) {

            if (args.length != 4) {

            System.out.printin("Usage:\tast input_file out_file
            summary_size"); else {

            String inFileName = argsl;

            String outFileName = args2;

            int summarysize = Integer.valueOf(args3);

            try {

                  String inText = readFil e(i nFil eName) ;
                  writeFile(outFileName, summarize(inText, summarysize, new
                  ArrayL-ist<ElementarryDiscourseUnit>()));,

            } catch (lOException e) { System.err.println("Error while
            reading/writing files...");

            }


      public static String summarizefString text, int summarysize,
      List<ElementarryDiscourseUnit> scoredUnits)

      {

            List<section> sections = parse(text);

            List<TreeAxiom> sectionTrees = new ArrayList<TreeAxiom>();
            List<TreeAxiom> paragraphTrees = new ArrayList<TreeAxiom>();
            List<List<TreeAxiom» sentenceTrees = new
            ArrayList<List<TreeAxiom»();

            for (section section : sections) {

                  Set<Relation> paragraphRelations =
                  relationHypothesis(section.getUnits());
                  sectionTrees.add(getValidTree(section.getunits(),
                  paragraphRelations));
```

```
            for (Paragraph paragraph : section.getParagraphes()) {
            Set<Relation> sentenceRelations =
            relationHypothesis(paragraph.getlinits());

      paragraphTrees.add(getvalidTree(paragraph.getunits(),
      sentenceRelations));

      List<TreeAxiom> sentences = new ArrayList<TreeAxiom>();     '

      for (Sentence sentence : paragraph.getSentencesO) {

      Set<Relation> clauseRelations = relationHypothe-
      sis(sentence.getunitsO);      '

      if (sentence.getUnits() .size() > 1)
      sentences.add(getvalidTree(sentence.getuni ts() , clauseRelati
      ons)); else { BinaryTree tree = new BinaryTreeO;

      ElementarryDiscourseUnit unit = sentence. getUnitsQ .get(O);

      tree.insert(unit.getindex(), RhetoricalRela-tion.LEAF,
      RhetoricalRelationstatus.NUCLEUS, unit, unit.getlndexO);

      sentences.add(new TreeAxiom(tree, unit.getlndexO,
      unit.getlndex(), new HashSet<Relation>()));

      sentenceTrees.add(sentences); }

      String summary = null;

      TreeAxiom mainTree = getMainTree(sectionTrees, paragraphTrees,
      sentenceTrees); if (mainTree != null) { summary =
      getSummary(mainTree, summarysize, scoredUnits);

      return summary;
      }

public static TreeAxiom getMainTree(List<TreeAxiom> sectionTrees,
List<TreeAxiom> paragraphTrees, List<List<TreeAxiom» sentenceTrees) {

assert(sectionTrees.size() == 1);

TreeAxiom sectionTree = sectionTrees.get(O);

int i = 0;

if (paragraphTrees.size() == sentenceTrees.size()) {

for (TreeAxiom paragraphTree : paragraphTrees) { paragraph-

Tree.updateLeafs(Axi omuti1s.getTreeNodes(sentenceTrees.get(i ++)));

} }

sectionTree.updateLeafs(Axiomuti1s.getTreeNodes(paragraphTrees));

List<ElementarryDiscourseUnit> list = new ArrayList<ElementarryDi
scourseuni t>(secti onT ree.getLeafs());

i = 1;
```

```
        for (ElementarryDiscourseUnit unit : list) unit. setlndex(i++);

        sectionTree.updatePromotion();

        sectionTree. setRight(I, st.size() ) ;

        return sectionTree;

        1

        public static void writeFile(string outFileName, String inText)

        throws lOException {

        Filewriter fw = null;

        try {

        fw = new Fi leWri ter(outFil eName) ;

        fw.write(inText);

        catch (lOException e) {

        System.err.println("Error while writing to file");

        finally {

        if (fw != null) { fw.close();}

}

        private static string getsummary(TreeAxiom tree, int summarysize,
        List<ElementarryDiscourseumt> resultscoredunits) {

              List<ElementarryDiscourseUnit> scoredunits = getscore(tree);

              for (ElementarryDiscourseUnit scoredunit : scoredunits) {
              System.out.print(scoredUnit.getlndex() + "->");

              Set<ElementarryDiscourseUnit> summary = SummarizationU-
              tils.summarizeText(summarySize, scoredunits);

              StringBuilder str = new StringBuilderO;

              for (ElementarryDiscourseUnit unit : summary) {
              str.append(unit.getTextO);

              resultscores.addAl1(scoredunits);

              return str.toStringO ;

        }

        public static TreeAxiom getValidTree(List<ElementarryDiscourseUnit>
        units, Set<Relation> relations) {

              if (units.size() == 1) return null;

              ProofTheoryAlgorithm theories = new ProofTheoryAlgorithm(units,
              relations); Set<Axiom> axioms = null;
```

```
        try { axioms = theories.execute();

        | catch (NoRelationsException e) {

        final int delta = units.get(O).getlndexQ - 1;

        Set<TreeAxiom> finalTrees = Axiomutils.findTree(axioms, 1 +
        delta, units.si ze() + del ta);

        TreeAxiom validTree = new
        StandartValidTreeFinderO.getValidTree(finalTrees);
        validTree.updateParent();

        return validTree;

    } .

    public static Set<Relation>
    relationHypothesis(List<ElementarryDiscourseUnit> units) {

        RelationHypothesesAlgorithm hypothesis = new
        RelationHypothesesAlgorithm(units); Set<Relation> relations =
        hypothesis.execute();

        WordCoOccurenceBasedHypothesizingAlgorithm extraHypothesis =
        new WordCoOccurence-
        BasedHypothesizinqAlgorithm(units.toArray(new
        ElementarryDiscourseUnit {}), relations);

        relations = extraHypothesis.executeO;

        return relations;

    }

    public static List<Section> parse(string text) {

        Parser parser = new ParserQ;

        List<Section> sections = parser.parseText(text);

        Textual UnitSegmentationAlgorithm segmentation = new Textual
        UnitSegmentationAlgo-rithm();

        for (Section section : sections) {

        for (Paragraph paragraph : section.getParagraphesO) {

        for (sentence sentence : paragraph.getSentencesQ) {
        segmentation.execute(sentence);

        sentence.setunits(new ArrayList<ElementarryDi
        scourseUnit>(segmentation.getclauses()));

        } }

        System.out.println("Found clauses:");

        for (Section section : sections) {

        for (Paragraph paragraph : section.getParagraphesO) {
```

```
            for (sentence sentence : paragraph.getSentences()) {

            System.out.println(sentence.toStringO);

            return sections;

      }

      private static List<ElementarryDiscourseUnit> getScore(TreeAxiom
      validTree) {

            validTree.score();

            List<ElementarryDiscourseUnit> list = new
            ArrayList<ElementarryDi scourseunit>(validTree.getLeafs());

            Col1ections.sort(1ist,
            ElementarryDiscourseUnit.EDU_SCORE_COMPARATOR); return list;

      }

      public static String readFile(String inFileName)

      throws lOException {

            Fileinputstream fin = null;

            BufferedReader reader = null;

            StringBuilder inText = new StringBuilderO;

            try {

            fin = new Fi 1 einputStream(inFileName) ;

            reader = new BufferedReader(new inputStreamReader(fin));

            String line = null;

            while ((line = reader.readLineO) != null) {

            inText.append(line);

} .
```

Додаток 3

Копія презентації

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

КАФЕДРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ КОМП’ЮТЕРНИХ СИСТЕМ

Метод автоматизованого реферування україномовних науково-технічних текстів

Доповідач: Левчук Ольга Сергіївна

Науковий керівник: к.т.н., доцент Заболотня Т.М.

Київ – 2021

АКТУАЛЬНІСТЬ ДОСЛІДЖЕННЯ

- Великі обсяги електронної науково-технічної інформації
- Відсутність систем, що використовують методи, що враховують нелінійну та ієрархічну природу тексту

Об’єкт дослідження: процес автоматизованого реферування текстових даних.

Предмет дослідження: методи та алгоритми автоматизованого реферування україномовних науково-технічних текстів.

ОКРЕМІ ЗАВДАННЯ

1. Проаналізувати існуючі методи автоматичного реферування тексту.

2. Розробити метод автоматичного реферування україномовних науково-технічних текстів.

3. Реалізувати систему автоматичного реферування тексту.

4. Оцінити результати роботи системи

ІСНУЮЧІ РІШЕННЯ

На сьогодні можна відокремити дві основних групи методів, призначених для задачі автоматизованого реферування тексту:

- екстрактивні методи;
- методи з опорою на знання.

Сучасні системи автоматичного реферування тексту:

- Intelligent Miner for Text
- MS Office AutoSummarize
- TextAnalyst
- Oracle Text
- Copernic summarizer

ПОРІВНЯННЯ ІСНУЮЧИХ РІШЕНЬ

Назва	Метод реферування	Підтримка української мови
Intelligent Miner for Text	Статистичні методи	Відсутня
MS Office Auto-Summarize	Статистичні методи	Присутня
TextAnalyst	Статистичні методи	Відсутня
Oracle Text	Статистичні методи	Присутня лише при завантаженні словників власноруч
Copernic summarizer	Статистичні та лінгвістичні методи	Відсутня

ГІПОТЕЗИ

При побудові тексту у вигляді дерева, базуючись на теорії риторичної структури, висувається наступне обмеження для визначення коректності структури тексту: якщо функціональне відношення лежить між двома елементами структури тексту, то воно лежить між, принаймні, двох ключових складових цих елементів

МЕТОД АВТОМАТИЗОВАНОГО РЕФЕРУВАННЯ УКРАЇНОМОВНИХ НАУКОВО-ТЕХНІЧНИХ ТЕКСТІВ

Для опису методу автоматизованого реферування україномовних науково-технічних текстів, нам необхідно більш детально розглянути його складові:

- формалізований опис структури україномовних науково-технічних текстів, що дозволить автоматизувати процес реферування тексту;
- визначення функціональних відношень між фрагментами тексту;
- побудова структури тексту на основі множини функціональних відношень між фрагментами тексту.

ФОРМАЛІЗОВАНИЙ ОПИС СТРУКТУРИ ТЕКСТУ

Підходом до опису тексту у запропонованому методі є теорія риторичної структури, в рамках якої стверджується, що текст може бути поданий у вигляді графу. При цьому, вузли графів будуть зв'язані один і тим самим набором відносин, які називаються функціональними відносинами, що означає, що кожна дискурсивна одиниця не існує самостійно, а додається до іншої для досягнення певної мети.

Відношення	Ядро	Сателіт
sentence	Перший елемент в перерахуванні	Другий елемент в перерахуванні
background	Текст, необхідний для розуміння	Текст для полегшення розуміння
elaboration	Базова інформація	Додаткова інформація

ТЕОРІЯ РИТОРИЧНОЇ СТРУКТУРИ

Основні положення:

- елементарні текстові елементи є непересічними між собою частинами тексту;
- функціональні відношення поєднують між собою частини тексту різного розміру;
- елементарні текстові елементи мають різну значущість у тексті;
- структура тексту може бути представленою у вигляді дерева;

Обмеження:

- функціональні структури є деревами, в яких елементи, що знаходяться на одному рівні являють собою неперервний текст;
- елементи можуть бути двох видів: ядро та сателіт;
- кожен текстовий елемент може бути пов'язаний з іншим лише одним функціональним відношенням;

ВИЗНАЧЕННЯ КРИТЕРІЮ КОРЕКТНОСТІ СТРУКТУРИ ТЕКСТУ

Висувається наступне обмеження для визначення коректності структури тексту: якщо функціональне відношення лежить між двома елементами структури тексту, то воно лежить між, принаймні, двох ключових складових цих елементів. З цього слідує декілька припущень:

- є необхідність роботи з розширеними функціональними відношеннями;
- є необхідність обробки невизначеностей.

Введемо наступні предикати для опису характеристик коректної структури:

- $isValidPosition(u_i, j)$
- $isInFunctionalRel(name, u_i, u_j$

ПРИКЛАД НЕВИЗНАЧЕНОСТІ

[Перша система призначена для обробки векторної графіки.]^{A}[Найкраще вона працює з зображеннями формату SVG.]^{B} [На відміну від першої системи, друга призначена для більшої кількості форматів.]^{C} [Особливо добре вона працює з форматами PNG та JPG.]^{D}

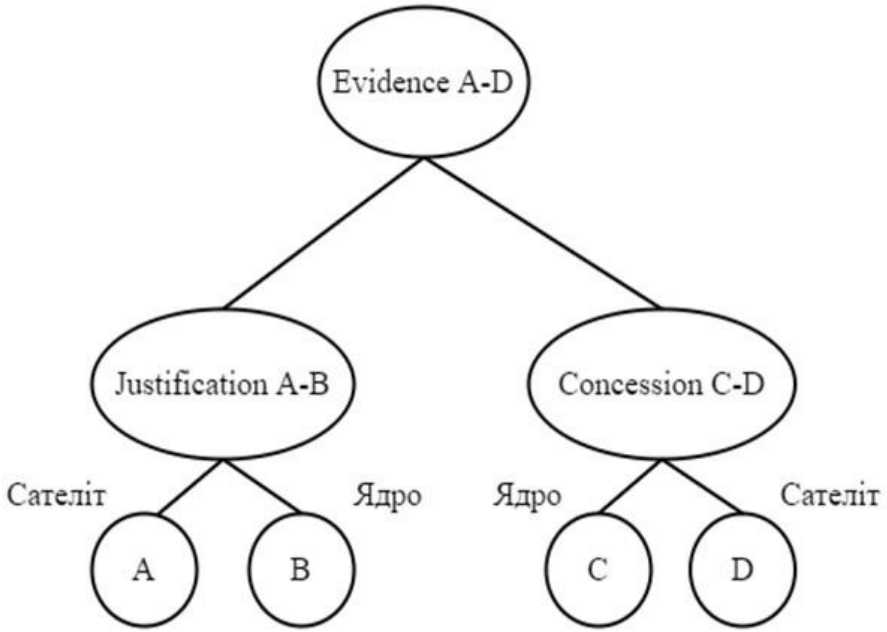
$$isInFunctionalRel(contrast, A, C) \oplus isInFunctionalRel(contrast, A, D) \oplus isInFunctionalRel(contrast, B, C) \oplus isInFunctionalRel(contrast, B, D)$$

ПРИКЛАД ПОБУДОВИ ДЕРЕВА

[Неважливо, наскільки складною є будь-яка програмна система,]^{A}[контроль якості є дуже важливою частиною її розробки.]^{B}[Відомо, що дуже велика кількість помилок припускається на етапі проектування,]^{C}[хоча розробники вважають, що всі вимоги до системи є зрозумілими.]^{D}

$$\begin{cases} isInFunctionalRel(justification, \mathrm{A}, \mathrm{B}) \\ isInFunctionalRel(justification, \mathrm{D}, \mathrm{B}) \\ isInFunctionalRel(evidence, \mathrm{C}, \mathrm{B}) \\ isInFunctionalRel(concession, \mathrm{D}, \mathrm{C}) \\ isInFunctionalRel(restatement, \mathrm{D}, \mathrm{A}) \\ isValidPosition(\mathrm{A}, 1) \\ isValidPosition(\mathrm{B}, 2) \\ isValidPosition(\mathrm{C}, 3) \\ isValidPosition(\mathrm{D}, 4) \end{cases}$$

ПРИКЛАД ПОБУДОВИ ДЕРЕВА



УЗАГАЛЬНЕНИЙ АЛГОРИТМ ПРОЦЕСУ АВТОМАТИЧНОГО РЕФЕРУВАННЯ ТЕКСТУ

14/28

АЛГОРИТМ ВИЗНАЧЕННЯ ФУНКЦІОНАЛЬНИХ ВІДНОШЕНЬ

Алгоритм визначення функціональних відносин базується на словнику ключових фраз та містить чотири етапи:

• розбиття тексту на речення та визначення для кожного з них набору ключових фраз або дискурсних маркерів;
• визначення меж елементарних текстових елементів;
• визначення функціональних відносин між елементарними текстовими елементами.

АЛГОРИТМ ВИЗНАЧЕННЯ ГРАНИЦЬ ЕЛЕМЕНТАРНИХ ТЕКСТОВИХ ЕЛЕМЕНТІВ

16/28

АЛГОРИТМ ПОБУДОВИ МНОЖИНИ ФУНКЦІОНАЛЬНИХ ВІДНОШЕНЬ НА ОСНОВІ НАБОРУ ЕЛЕМЕНТАРНИХ ТЕКСТОВИХ ЕЛЕМЕНТІВ

17/28

АЛГОРИТМ ПОБУДОВИ МНОЖИНИ ФУНКЦІОНАЛЬНИХ ВІДНОШЕНЬ ДЛЯ НЕПОВ'ЯЗАНИХ ЕЛЕМЕНТАРНИХ ТЕКСТОВИХ ЕЛЕМЕНТІВ

18/28

АЛГОРИТМ ОТРИМАННЯ РЕФЕРАТУ

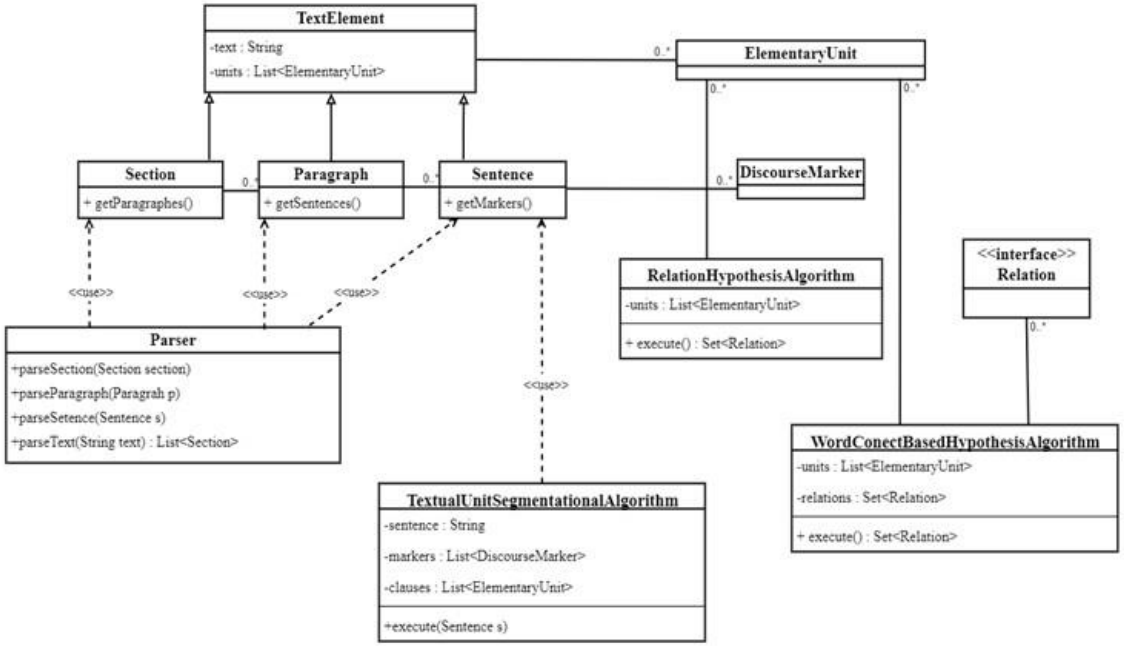
Вхідні дані: текст T, число p що є обсягом реферату у відсотках, від 1 до 100.

Вихідні дані: набір p% найбільш значущих елементарних текстових елементів.

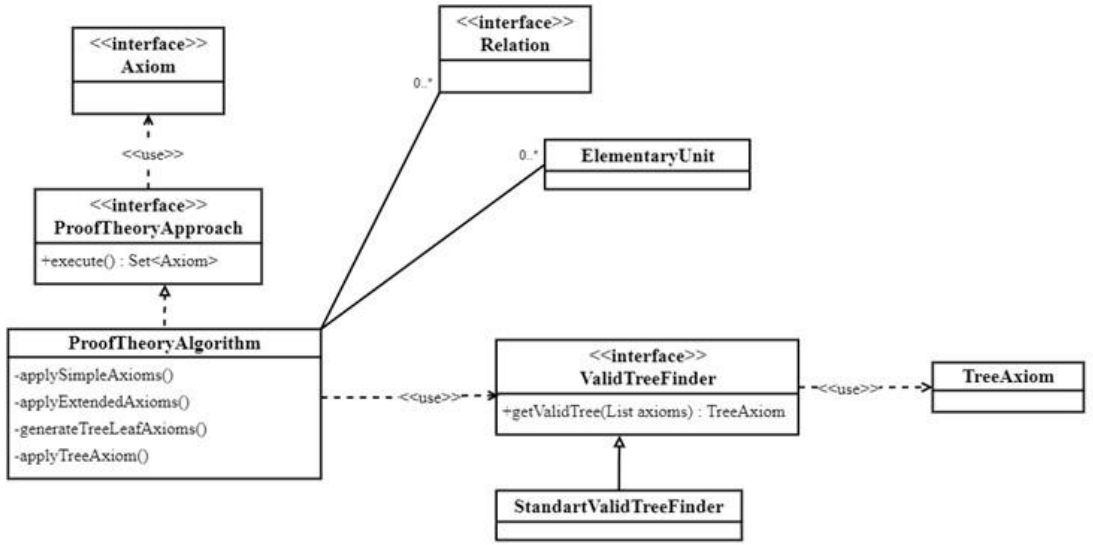
- Визначити структуру тексту за допомогою алгоритму побудови структури тексту.
- Відсортувати список елементарних текстових елементів за їх значущістю.
- Обрати необхідний відсоток елементарних текстових елементів з початку даного відсортованого списку для створення реферату.

СТРУКТУРА СИСТЕМИ АВТОМАТИЗОВАНОГО РЕФЕРУВАННЯ

ДІАГРАМА КЛАСІВ ПІДСИСТЕМИ АНАЛІЗУ ТЕКСТУ



ДІАГРАМА КЛАСІВ ПІДСИСТЕМИ ПОБУДОВИ КОРЕКТНИХ СТРУКТУР ТЕКСТУ



ОЦІНЮВАННЯ ЕФЕКТИВНОСТІ

Результати оцінки розробленого алгоритму на основі експертних оцінок

Елементарний текстовий елемент	Експерти					Система
	1	2	3	4	5	
1	0	1	0	1	0	3
2	2	2	2	2	2	6
3	1	1	1	2	1	4
4	0	1	0	0	0	3
5	1	1	0	0	0	3
6	0	0	0	0	0	1
7	0	1	1	0	0	3
8	2	2	2	2	2	5
9	0	1	1	0	0	3
10	0	1	1	1	1	4

ОЦІНЮВАННЯ ЕФЕКТИВНОСТІ

Середні значення показників якості методів автоматизованого реферування

Система (метод)	Повнота	Точність	F-параметр
Розроблена система (метод на основі аналізу функціональних відношень)	0.6481	0.6703	0.6603
СистемаMS Office AutoSummarize (метод на основі підрахунку статистичних показників)	0.3518	0.3275	0.3392

ОЦІНЮВАННЯ ЕФЕКТИВНОСТІ

Залежність якості реферату від обсягу тексту

Текст	Кількість слів	Кількіть речень	Кількість ключових фраз	Коефіцієнт зустрічаємості ключових фраз	Повнота	Точність	F-параметр
1	130	8	2	0.25	0.386	0.412	0.3984
2	150	10	3	0.3	0.431	0.446	0.4383
3	170	12	4	0.33	0.451	0.555	0.4978
4	230	16	7	0.44	0.581	0.565	0.5738
5	295	21	12	0.57	0.623	0.634	0.6286

ВИСНОВКИ

- На основі аналізу існуючих підходів автоматичного реферування текстів, можна зробити висновок, що методи з опорою на знання майже не використовуються, оскільки недостатньо досліджені для опису властивостей тексту. В результаті проведеного аналізу було сформульовано мету та основні задачі роботи.
- Розроблений метод формалізованого опису структури тексту, що базується на використанні теорії риторичних структур, яка допоможе врахувати нелінійну та ієрархічну природу тексту, що дозволяє підвищити якість автоматичного реферування україномовних науково-технічних текстів. Метод формалізованого опису містить визначення характеристик структур тексту та обмежень, що накладаються на коректні структури тексту.
- Розроблений алгоритм визначення функціональних відносин між фрагментами тексту, що базується на використанні вузькоспеціалізованого словника ключових фраз української мови, що значно зменшує надлишковість інформації, що виникає при використанні словників загального призначення.

ВИСНОВКИ

- Розроблений алгоритм побудови структури тексту на основі множини визначених функціональних відношень між фрагментами тексту, який відрізняється врахуванням неоднозначності відношень з ключових фраз, генерацією альтернативних множин варіантів коректних структур тексту та вибір оптимальної за критерієм сукупної метрики.
- Проведена експериментальна перевірка запропонованих методів та алгоритмів, що реалізовані в розробленому програмного забезпечення автоматизованого реферування україномовних науково-технічних текстів. Згідно з проведеними дослідженнями, можна стверджувати що якість є в середньому на 20% вищою у порівнянні с рефератами, отриманими за допомогою традиційного методу, реалізованого у вбудованій функції пакету MS Office – AutoSummarize

Дякую за увагу!

Питання?